2 (2), 2021

https://dergipark.org.tr/en/pub/jscai

Editorial Board

Contents

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:56-68 (2021),
Received: May 8, 2021 Accepted: July 4, 2021
JSCAI
ISSN: 2717-8226

Deep hybrid models for CT images to detect COVID-19: A comparison of transfer learning approach

[1]Ebru ERDEM , [2]Tolga AYDIN

[1] Department of Computer Engineering, Ataturk University, 25240, Erzurum, Turkey
[2] Department of Computer Engineering, Ataturk University, 25240, Erzurum, Turkey

Corresponding author: Ebru ERDEM (e-mail: ebruerdem@atauni.edu.tr).

ABSTRACT The COVID-19 has become a pressing public health concern recently due to its dramatic impact. It spreads quickly, and it is beyond the ability of health staff to detect patients with the disease immediately. However, the ability to diagnose SARS-CoV-2 in a short time is critical for fighting the disease. The primary objective of this study is to develop deep neural networks to diagnose disease in a quick, safe, and cheap way. We classify the cases as normal, COVID-19, and pneumonia. Deep neural networks are developed to perform a three-class classification task. Ten deep learning models are evaluated on a large dataset. Although all DCNNs demonstrated promising potential for classification, hybrid neural networks delivered the most promising outcome with the highest accuracies. The first hybrid model is named MICOVID. The second hybrid model is named VVCOVID. These models are developed through transfer learning by using pre-trained deep learning models. Performance metrics results showed that MICOVID and VVCOVID models have an accuracy of 94% for COVID-19 detection. This is higher than other classification models. These findings suggest that two novel hybrid models that we proposed have great potential to be embedded into computer-aided systems to predict disease in radiology departments.

KEYWORDS: CNN, COVID-19, deep neural network, hybrid models

1. INTRODUCTION

Since 2019, SARS-CoV-2 has spread from China to other countries. Coronavirus disease 2019 (COVID-19) has quickly become a pandemic [1]. By April 2020, more than 30 million were confirmed as coronavirus cases, 28 million recovered, and more than 1 million deaths were reported [2]. Early diagnosis is critical to prevent infection to healthy people. The reverse-transcription polymerase chain reaction (RT-PCR) is the standard technique for diagnosing the disease. However, it may give false-negative results in the early stages of diseases. By comparison, CXR imaging or CT imaging technique is more helpful for COVID-19 detection. Bilateral and peripheral predominant ground-glass opacities (GGO) in the lobes are common initial findings on CT images. Besides bilateral multifocal GGO, septal thickening and pleural thickening are other common manifestations in the later stages [3, 4]. At the early stage of disease, it may be hard to view GGO. Therefore, images must be interpreted by only expert radiologists. Early diagnosis is getting more difficult in the face of the immense amount of the suspected cases and a limited number of expert radiologists. Computer-aided diagnosis systems (CAD) are necessary to solve such problems. Artificial intelligence solutions provide powerful tools for overcoming these difficulties.

Deep learning (DL) approaches are considered as the sub-class of machine learning (ML) methods. These methods have been used widely in the field of medicine. DL methods are capable of feature extraction and representation from raw data without any hand-craft method, augmentation, or segmentation. This situation provides an advantage compared to ML methods (thanks to improved diagnosis or smooth classification process, shorter time, and less cost). For controlling and combating novel coronavirus, DL methods can provide a quick and efficient solution. DL-based models have shown high performance in detecting COVID-19 [5- 9].

Compared to other techniques, Chest CT is considered to have sensitivity for clinical findings [10, 11]. Abbasian Ardakani et al. (2020) proposed an AI-based method for radiologists to improve the diagnosis of COVID-19. In their study, ten convolutional neural networks were used (AlexNet, VGG-16, VGG-19, ResNet-18, ResNet-50, ResNet-101, Xception, GoogleNet, MobileNet-V2, and SqueezeNet). The best performance was obtained by Xception (99.02% accuracy rate) and ResNet-101 (99.51% accuracy rate). Also, as a highly sensitive model, ResNet-101 [12] can be considered. Zhang et al. proposed another AI system developed on CT database from 3.777 patients. The AI system performance was tested by using U-net, DRUNET, FCN, DeepLabv3 (segmentation frameworks). For an accurate diagnosis of NCP, 40.880 slices were used from 260 patients (including 83 NCP patients, 86 normal patients, and 91 pneumonia patients) as a test to the classifier model. A 92.49% accuracy, a 94.93% recall, a 91.13% specificity, and a 97.97% AUROC [13] were achieved by using the proposed system. Attallah et al. constructed an efficient MULTI-DEEP CAD system based on multiple CNNs for detecting the disease. The CAD system comprises different four scenarios. Scenario I is composed of four pre-trained CNNs for classification (accuracy (78.29%). In scenario II, features were extracted from pre-trained CNNs (AlexNet, GoogleNet, ShuffleNet, and ResNet-18), and SVM was used as a classifier (accuracy (92.5%). For scenario III, the principal component analysis was applied to each feature extracted from pre-trained CNN, and selected principal components were used to train the SVM classifiers (accuracy (94%). In the last scenario, for capable of with compare scenario III, the four features were extracted from pre-trained CNN, and these features were used to train the SVM classifier (The accuracy is 94.7%). This verified system detected COVID-19 with high accuracy [14]. In another article, 386 Covid-19 and 1010 Non-Covid-19 CT lung images [15] were studied. Data were augmented with noise-adding, distortion, brightness, and contrast-changing methods. Within the scope of the study, 23-layer CNN architecture was proposed for classification. The study achieved the best accuracy rate of 93.94% and 95.70%, for 2-fold cross-validation and 10-fold cross-validation, respectively. Stephanie et al. obtained 90.8% accuracy, 93% specificity, and 84% sensitivity for the classification of 1337 patients with deep learning algorithms (Grad-CAM method) [16]. Jaiswal et al. used DenseNet-201 based deep transfer learning to detect NCP on SARS-CoV-2 CT scan images with AUC of 97%, an accuracy of 96.25%, a specificity of 96.21%, an F-measure of 96.29%, a recall of 96.29%, and a precision of 96.29% [17]. Li et al. developed a DL-based method to analyse the NCP from thick-section CT scans. The DL-based method has the ability to obtain good results with an AUC of 96.8% [18]. M. Hasan et al. built a feature extraction method comprising of DL and Q-deformed entropy algorithm.

These features were classified by employing LSTM for NCP, pneumonia, and normal cases with an accuracy of 99.68% [19]. Ko et al. utilized a 2D deep learning framework to COVID-19 pneumonia diagnosis on the lung CT image. This framework was named FCONet. FCONet framework use Xception, Inception-v3, ResNet-50, and VGG16 pre-trained models as its backbone. It showed excellent performance with a sensitivity of 99.58%, an accuracy of 99.87%, and a specificity of 100% [20].

DL-based approaches applied for automated detection of COVID-19 are summarized in Table 1 (using CXR or CT images). The CNN-based frameworks for novel coronavirus pneumonia (NCP) diagnosis were suggested in other studies. But, in usual, the effectiveness of concatenating deep models were not considered in previous research studies. To address this void in the literature, two novel hybrid deep neural networks are proposed in this article. The first hybrid model integrates MobileNet and Inception-V3 architectures (MICOVID). The second hybrid model integrates VGG16 and VGG19 architectures (VVCOVID). We also compared our proposed hybrid models with other deep learning models (DPN98, Xception, Inception-ResNet-V2, MobileNet, Inception-V3, VGG16, VGG19, SqueezeNet) in terms of model accuracy, precision, recall, f1-score, and confusion matrix. All these models were evaluated on 2019nCoVR, a large public dataset of the China National Center for Bio-information.

Table 1. A summary of previous research studies

Paper	Dataset	Method	Results
Attallah et al. [14]	347 COVID-19 397 non-COVID-19	ResNet-18	Accuracy (78.29%) AUC (83.82%) Sensitivity (76.9%) Specificity (79.9%) Precision (81%) F1-score (78.9%)
Ahuja et al. [21]	178 COVID-19 228 non-COVID-19	ResNet-18	Accuracy (99.4%) AUC (99.65%) Sensitivity (100%) Specificity (98.6%)
Dansana et al. [22]	360 COVID-19 34 non-COVID-19	VGG16	Accuracy (91%) Sensitivity (94%) Precision (100%) F1-score (97%)
Panwar et al. [23]	192 COVID-19 145 non-COVID-19	nCOVnet	Accuracy (97.62%) Sensitivity (97.62%) Specificity (78.57%)
Karar et al. [24]	69 COVID-19 237 non-COVID-19	ResNet50V2 and VGG16	Accuracy (99.90%)
Mamunur et al. [25]	260 COVID-19 600 non-COVID-19	VGG19	Accuracy (89.3%) Sensitivity (89%) Precision (90%) F1-score (90%)
Elasnaoui et al.[26]	1493 COVID-19 4594 non-COVID-19	Inception-ResNetV2	Accuracy (92.18%) Sensitivity (92.11%) Specificity (96.06%) Precision (92.38%)

			F1-score (92.07%)
Abbasian Ardakani [12]	108 COVID-19 912 non-COVID-19	ResNet-101	Accuracy (99.51%) AUC (99.4%) Sensitivity (100%) Specificity (99.02%)
Abbasian Ardakani [12]	108 COVID-19 912 non-COVID-19	Xception	Accuracy (99.02%) AUC (99.4%) Sensitivity (98.04%) Specificity (100%)
Jaiswal et al. [17]	1262 COVID-19 1230 non-COVID-19	DenseNet-201	Accuracy (96.25%) Sensitivity (96.29%) Specificity (96.21%) Precision (96.29%) F1-score (96.29%)

The general flow diagram of the study is given in Fig. 1. This study aims to develop deep learning algorithms for predicting disease in real-time. So, safety mechanisms can be utilized to prevent death and serious consequences. A deep learning framework was employed to predict disease from CT images. We use CNN features (deep learning framework) that directly predict from raw data without any need for hand-crafted features, unlike classical machine learning approaches for feature extraction (hand-crafted).

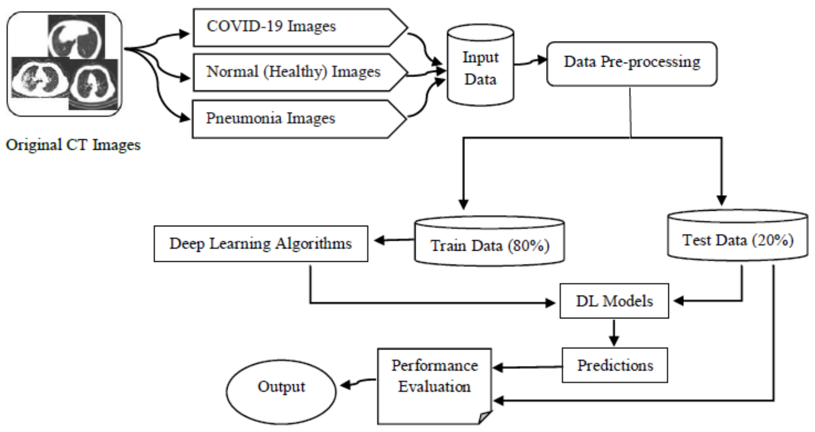

Figure 1. Flow diagram of the study

2. MATERIALS AND METHODS

2.1 DATASET

The 2019nCoVR dataset consists of CT images (2019nCoVR database, which is available at http://ncov-ai.big.ac.cn/download?lang=en). These images are constructed from the China Consortium of Chest CT Image Investigation, and they are classified as SARS-CoV-2 virus, pneumonia, and normal. This dataset is publicly available with the aim to combat disease. 31 files containing CT scans of COVID-19, 32 files containing CT scans of pneumonia, and 27 files containing CT scans of normal have been structured in the dataset [27]. We prepared a subset of CT images from the 2019nCoVR dataset. The combined dataset consists of 137.263 CT images, 114.416 training, and 22.847 testing samples. This dataset consists of 50.145 CT scans of COVID-19, 30.014 CT scans of pneumonia, and 34.257 CT scans of normal. The dataset is organized in 2 folders (train, test), as given in Table 2. CT images are resized to 256x256 before fed to DNN.

Table 2. General distribution of dataset in this study

Dataset	COVID-19	Pneumonia	Normal (Healthy)
2019nCoVR [1]	50.145	30.014	34.257
Training Set	40.114	24.011	27.444
Test Set	10.031	6.003	6.813

2.2 THE PROPOSED FRAMEWORK

In this study, ten deep learning models are applied to perform the classification. These models are a DPN98 [28] model, an Xception [29] model, an Inception-ResNet-V2 [30] model, a MobileNet [31] model, an Inception-V3 [32-34] model, a VGG16 [35] model, a VGG19 [35] model, a SqueezeNet [36] model and two hybrid models (MICOVID and VVCOVID) (Fig. 2). Transfer learning with popular pre-trained deep neural networks (DPN98, Xception, Inception-ResNet-V2, MobileNet, Inception-V3, VGG16, VGG19, and SqueezeNet) are applied to the training images of the 2019nCoVR dataset.

2.2.1 TRANSFER LEARNING APPROACH

Transfer learning is used in the field of deep-learning as a basic method. It allows a model trained on one task to be repurposed to another task through adopting. Thus, it utilizes previously learned knowledge in new classification problems. Consequently, it is able to obtain faster and better results. This approach is very useful for medical image classification.

Pre-trained models can be used for different tasks. In this study, they are used as a feature extractor (pre-trained CNN features). For classification, a classifier is trained on top of the pre-trained models. For this reason, we only fine-tune the last layer of deep neural networks (as three-class). Eight popular pre-trained models were evaluated: DPN98, Xception, Inception-ResNet-V2, MobileNet, Inception-V3, VGG16, VGG19, and SqueezeNet.

2.2.2 CNN

CNN architecture is composed of different layers: convolution, pooling and fully connected ones. The convolution layer is applied to input data (images, etc.) to produce a feature map by convolution filter (kernel). After the convolution layer, the pooling layer is generally applied to reduce the dimensions (to provide a down-sample for each feature map, to reduce the number of parameters). The output feature map and pooling process are obtained in Equation (1) and Equation (2), respectively. In Equation (1), $\boldsymbol{X_i^{l-1}}$, $\boldsymbol{k_{ij}^l}$,$\boldsymbol{b_j^l}$, and f() represent the local features, kernel (filter), bias, and activation function, respectively. In Equation (2), down() represents down-sampling process of pooling layer.

$$X_j^l = f\left(\sum\nolimits_{i \in M_j} X_i^{l-1} * k_{ij}^l + b_j^l\right) \quad (1)\ [37]$$

$$X_j^l = down\left(X_j^{l-1}\right) \quad (2)\ [38]$$

Finally, a fully connected layer (the last layer) takes the output of previous layers (convolution or pooling) and predicts the label of input data for the classification decision.

2.2.3 COVID-19 DETECTION USING DPN98, XCEPTION, INCEPTION-RESNET-V2, MOBILENET, INCEPTION-V3, VGG16, VGG19, AND SQUEEZENET

Transfer learning on a dataset is used to train eight convolutional neural networks, including SqueezeNet, VGG19, VGG16, Inception-V3, MobileNet, Inception-ResNet-V2, Xception, and DPN98 to identify disease in 114.416 CT images. We evaluated these models on 22.847 CT images. So, instead of building a CNN architecture, we used some pre-trained CNN models. The detailed information about pre-trained models are as follows:

. Two of the models are the pre-trained VGG16 and VGG19, trained on the ImageNet dataset. VGG16 and VGG19 obtained good results in the ILSVRC-2014. These networks are a subset of the VGG network. VGG16 architecture consists of 16 convolutional layers. VGG19 architecture consists of 19 convolutional layers. These models are deeper CNN architectures. For both models, the default image input sizes are 224*224 with three channels.

. SqueezeNet model is a CNN architecture. The model has a convolution layer (independent, conv1), fire modules (eight, fire2-9), and the last convolution layer. Although 50 X smaller the size of this model compared to AlexNet, it gives higher performance. For this model, the dimensions of images in the input layer are 224 x 224 x 3.

. Inception-V3 CNN architecture was introduced by Szegedy et al. in 2015. The model was first introduced as a structure consisting of 22 layers (Inception-V1-GoogleNet). Later, the model was reintroduced as Inception-V2 (with batch normalization), and the final iteration has been referred to as Inception-V3 (with additional factorization). In addition Inception-V1 and Inception-V2, the Inception-V3 network uses a splitting method for dividing volume integrals. The Inception module, as a typical CNN architecture, contains convolution and pooling layers. For this model, the dimensions of images in the input layer are 229 x 229 x 3.

. Inception-ResNet-V2 is a hybrid network combining Inception architecture and residual connections [39]. Inception-ResNet-V2 is a residual version of Inception, which is roughly the computational cost of the Inception-v4 network. While Inception networks utilize filter concatenation, Inception-ResNet networks utilize residual connections.

. MobileNet is a popular deep learning model. While this CNN model has fewer parameters, it also has less calculation cost because of its general construction. Based on the depth-wise separable convolution, it has 28 layers. Except for the final layer, all layers are followed by a ReLU and a batch norm layer in the architecture. The model is designed for mobile devices and shows excellent performance for embedded industrial equipment.

. Xception was proposed by Francois Chollet. This model is another improvement of Inception-V3 deep learning model. The model contains depthwise separable convolutional layers with residual connections. For this model, image dimensions in the input layer are 229 x 229 x 3.

. DPN98 is a variation of Dual-Path Network (DPN) proposed by Yunpeng Chen, etc. It is a family of CNNs, and it has cost about fewer parameters than ResNeXt-101. DPN has the advantages of DenseNet (exploring new features) and ResNet (feature reuses). This model is very suitable for optimization and classification. It is especially used for image classification, segmentation, and object detection. DPN showed high accuracy on Places365-Standard and ImageNet-1k datasets [28]. The model has obtained the best results in the ILSVRC-2017 Challenge.

2.2.4 COVID-19 DETECTION USING MICOVID AND VVCOVID

As hybrid models, we present two novel concatenated CNN based approaches for COVID-19 detection in this study. The first proposed hybrid model is a combination of MobileNet and Inception-V3. This model is named MICOVID in the study. The second proposed hybrid model is a combination of VGG16 and VGG19. This model is named VVCOVID in the study. The proposed hybrid models have five steps as shown in Fig. 2.

These steps are as follows:

. CT scan images have been reshaped to 256*256 with three channels.

. The learned weights of the pre-trained models are used on ImageNet (for the first hybrid model: MobileNet, Inception-V3, and for second hybrid model: VGG16, VGG19). Thus, features are extracted automatically for each CT image.

. The obtained features are concatenated after applying "*GlobalAveragePooling Layer*".

. We use the stacking model to concatenate the output shape of models for each hybrid model through "Concatenate Layer". The concatenate layer returns a single tensor.

. Finally, two "*Fully Connected Layers*" consisting of 512 neurons are used. In the output layer, classification is made with the "*Softmax Layer*".

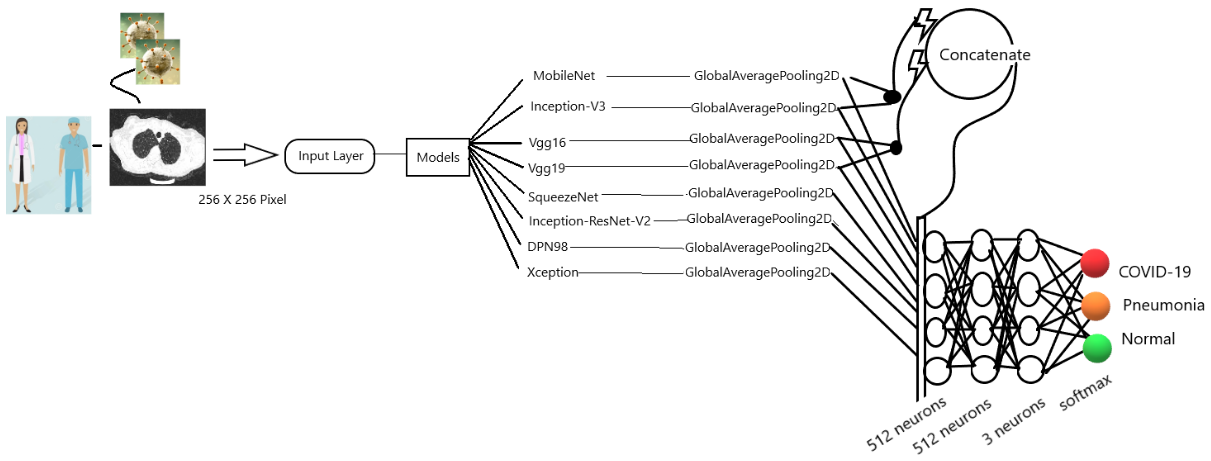

Figure 2. A general illustration of proposed framework.

MICOVID and VVCOVID layer, output shape and parameter details are given in Table 3.

Table 3. Proposed hybrid models details for detection of COVID-19

Model	Layer	Output Shape	Parameter	Total Parameter
MICOVID	input_1	(None,256,256,3)	0	
	mobilenet_1.00_224	(None,8,8,1024)	3228864	
	inception_v3	(None,6,6,2048)	21802784	
	global_average_pooling_2d_1	(None,1024)	0	
	global_average_pooling_2d_2	(None,2048)	0	26,869,219
	concatenate_1	(None,3072)	0	
	dense_1	(None,512)	1573376	
	dense_2	(None,512)	262656	
	dense_3	(None,3)	1539	
VVCOVID	input_1	(None,256,256,3)	0	
	vgg19	(None,8,8,512)	14714688	
	vgg16	(None,8,8,512)	21802784	
	global_average_pooling_2d_1	(None,512)	0	
	global_average_pooling_2d_2	(None,512)	0	35,528,067
	concatenate_1	(None,1024)	0	
	dense_1	(None,512)	1311232	
	dense_2	(None,512)	262656	
	dense_3	(None,3)	1026	

3. EXPERIMENTAL RESULTS

All model implementations and evaluations are done in Keras [40] by using a computer having an Intel(R) Core(TM) i7-7700 CPU, 24 GB memory and GeForce GT 730 GPU (NVIDIA).

3.1 DEEP LEARNING MODELS HYPER-PARAMETERS

Deep learning networks are trained for 500 epochs. The batch size is set to 32, an ADAM optimizer function and a learning rate of 0.0001. Trainable parameter, non-trainable parameter and total parameter size of DCNN models are given Table 4.

Table 4. Trainable parameter, non-trainable parameter and total parameter size of DCNN models in this study

Model	Trainable parameter	Non-trainable parameter	Total parameter
Inception-V3	23,081,635	34,432	23,116,067
MobileNet	3,995,971	21,888	4,017,859
Vgg16	15,241,539	0	15,241,539
Vgg19	20,551,235	0	20,551,235
SqueezeNet	1,249,347	0	1,249,347
InceptionResNetV2	55,327,331	60,544	55,387,875
DPN98	60,654,435	148,320	60,506,115
Xception	22,120,235	54,528	22,174,763
MobileNet+Inception-V3	26,812,899	56,320	26,869,219
Vgg19+Vgg16	35,528,067	0	35,528,067

3.2 EVALUATION METRICS

For each of the different DL models, we evaluated the classification performance in terms of sensitivity (recall), precision, accuracy, and F1-score. Table 5 and Table 6 summarize the classification performances of DCNN models. As seen in the tables, DCNN models give promising results for COVID-19 diagnosis. VVCOVID and MICOVID obtained the best accuracy rate of 94%. These hybrid models showed better performance than other DCNN models in terms of accuracy rate. VVCOVID, MICOVID, MobileNet, Inception-V3, VGG16, SqueezeNet, Inception-ResNet-V2, and Xception models obtained the sensitivity rate of ~95% ± 2%. Each model has achieved around 90% precision and F1-score rates. DCNNs have achieved the specificity rate of ~92% ± 6.7%. For COVID-19 diagnosis, the results have shown that Inception-ResNet-V2 model had the poorest classification performance with a specificity rate of 91.73%.

Table 5. Classification results of pre-trained models

	Class	MobileNet	Inception-V3	VGG16	VGG19	SqueezeNet	Inception-ResNet-V2	DPN98	Xception
Sensitivity (%)	Covid	97	92	94	87	96	97	75	96
Precision (%)	Covid	98	94	97	90	94	90	98	94
Accuracy (%)	Covid	93	86	92	86	93	89	71	93
F1-score (%)	Covid	98	93	95	89	95	94	85	95
Specificity (%)	Covid	98.51	95.79	98.55	92.22	95.27	91.73	98.75	95.27

Table 6. Classification results of deep hybrid models

	VVCOVID	MICOVID
Sensitivity (%)	96	97
Precision (%)	96	97
Accuracy (%)	**94**	**94**
F1-score (%)	96	97
Specificity (%)	97.77	97.09

The confusion matrix is a heuristic metric used to obtain the sensitivity (TPR), specificity (TNR), precision (PPV), accuracy (ACC), and F1-scores of the model, which are described in Equation (3), Equation (4), Equation (5), Equation (6), and Equation (7). Table 7 shows true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) based on three-class (COVID-19, pneumonia, and normal). So, we expressed performance metrics with subscripts for COVID-19 class as "*covid*", for pneumonia class as "*pneumonia*" and the healthy (normal) class as "*normal*". For example, TP_{covid} is the number of COVID-19 testing data correctly classified. TN_{covid} is the number of non-COVID-19 testing data correctly classified. FN_{covid} is the number of non-COVID-19 testing data misclassified. FP_{covid} is the number of COVID-19 testing data misclassified. $TP_{pneumonia}$ is the number of pneumonia testing data correctly classified. $TN_{pneumonia}$ is the number of non-pneumonia testing data correctly classified. $FN_{pneumonia}$ is the number of non-pneumonia testing data misclassified. $FP_{pneumonia}$ is the number of pneumonia testing data misclassified. TP_{normal} is the number of normal testing data correctly classified. TN_{normal} is the number of non-normal testing data correctly classified. FN_{normal} is the number of non-normal testing data misclassified. FP_{normal} is the number of normal testing data misclassified.

$$Sensitivity = \frac{\mathrm{TP}}{\mathrm{TP + FN}} \qquad (3)$$

$$Specificity = \frac{\mathrm{TN}}{\mathrm{TN + FP}} \qquad (4)$$

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP + FP}} \qquad (5)$$

$$Accuracy = \frac{\mathrm{TP + TN}}{\mathrm{TP + TN + FP + FN}} \qquad (6)$$

$$F1 - score = \frac{\text{2 x precision x sensitivity}}{\mathrm{TP + TN + FP + FN}} \qquad (7)$$

Table 7. Confusion matrix of the observed deep learning algorithms on 2019nCoVR dataset

DCNNs	**Actual Classes**	**Predicted**		
		Covid	pneumonia	normal
MobileNet	Covid	9763 (TP)	252 (FN)	16 (FN)
	pneumonia	130 (FP)	6679 (TN)	4 (TN)
	normal	60 (FP)	1107 (TN)	4836 (TN)
Inception-V3	Covid	9254 (TP)	376 (FN)	401 (FN)
	pneumonia	515 (FP)	6192 (TN)	106 (TN)
	normal	24 (FP)	1742 (TN)	4237 (TN)
VGG16	Covid	9391 (TP)	572 (FN)	68 (FN)
	pneumonia	133 (FP)	6649 (TN)	31 (TN)
	normal	123 (FP)	945 (TN)	4935 (TN)
VGG19	Covid	8755 (TP)	710 (FN)	566 (FN)
	pneumonia	582 (FP)	5948 (TN)	283 (TN)
	normal	415 (FP)	714 (TN)	4874 (TN)
SqueezeNet	Covid	9603 (TP)	308 (FN)	120 (FN)
	pneumonia	471 (FP)	6199 (TN)	143 (TN)
	normal	134 (FP)	410 (TN)	5459 (TN)
Inception-ResNet-V2	Covid	9766 (TP)	168 (FN)	97 (FN)
	pneumonia	691 (FP)	6086 (TN)	36 (TN)
	normal	368 (FP)	1103 (TN)	4532 (TN)
DPN98	Covid	7562 (TP)	48 (FN)	2421 (FN)
	pneumonia	85 (FP)	3118 (TN)	3610 (TN)
	normal	75 (FP)	359 (TN)	5569 (TN)
Xception	Covid	9603 (TP)	308 (FN)	120 (FN)
	pneumonia	471 (FP)	6199 (TN)	143 (TN)
	normal	134 (FP)	410 (TN)	5459 (TN)
VVCOVID	Covid	9660 (TP)	227 (FN)	144 (FN)

	pneumonia normal	317 (FP) 55 (FP)	6331 (TN) 402 (TN)	165 (TN) 5546 (TN)
MICOVID	Covid pneumonia normal	9715 (TP) 236 (FP) 49 (FP)	298 (FN) 6557 (TN) 685 (TN)	18 (FN) 20 (TN) 5269 (TN)

4. DISCUSSION-CONCLUSIONS

Chest CT may play an important role, especially where the PCR resulting in false-negatives in COVID-19 diagnosis. Since a limited number of CXR images is publicly available for COVID-19, besides CXR images, we have resorted to CT images . We have reported a framework comprising of DCNN models for COVID-19 detection from CT images. Also, two novel deep hybrid models are developed using the images obtained. These models were trained and tested on the 2019nCoVR sub-dataset (114.416 training set and 22.847 testing set). To report a summarizing performance of deep learning models, we provide the confusion matrix, accuracy, precision, sensitivity, specificity, and f1-scores for each of these models. When the results are generally examined, this proposed framework promises using CT scan images for disease diagnostics. Besides pre-trained models, with the novel proposed deep hybrid models, the sensitivity, specificity, accuracy, precision, and F1-scores for classifying COVID-19 are obtained, and these are ~96%, ~97%, ~94%, ~96%, and ~96%, respectively. The best accuracy rate of 94% is obtained with MICOVID and VVCOVID. MobileNet- SqueezeNet-Xception (93%), VGG16 (92%), Inception-ResNet-V2 (89%), Inception-V3-VGG19 (86%), and DPN98 (71%) follows it.

Because of the mounting amount of COVID-19-infected patients, health staff is having more difficulty in combating the disease. So, the rapid development of AI methods is critical. Consequently, DCNN based-CAD systems are recommended for the diagnosis of COVID-19 in this paper. These proposed models deliver faster results than the classical PCR testing method. Thus, in addition to pre-trained models, MICOVID and VVCOVID can become rapid and efficient diagnostic tools for COVID-19.

REFERENCES

[1] Anonim, “World Health Organization”, [Çevrimiçi]. Available at: https://www.who.int/emergencies/diseases/novel-coronavirus-2019, (accessed 28 September 2020).

[2] Anonim, “worldometer”, [Çevrimiçi]. Available at: https://www.worldometers.info/coronavirus/, (accessed 28 September 2020).

[3] A. Cinkooglu, H. A. Esmat, S. Recep, and M. N. Forogh, “COVID-19 presenting with a small ground-glass opacity in the upper lobe of the lung,” *Eurorad*, 2020.

[4] M. Parekh, A. Donuru, R. Balasubramanya, and S. Kapur, “Review of the chest CT differential diagnosis of ground-glass opacities in the COVID era,” *Radiology*, 202504, 2020.

[5] S. Minaee, R. Kafieh, M. Sonka, S. Yazdani, and G. J. Soufi, “Deep-covid: Predicting covid-19 from chest x-ray images using deep transfer learning,” *arXiv preprint arXiv:2004.09363*, 2020.

[6] H. Panwar, P. K. Gupta, M. K. Siddiqui, R. Morales-Menendez, P. Bhardwaj, and V. Singh, “A deep learning and grad-CAM based color visualization approach for fast detection of COVID-19 cases using chest X-ray and CT-Scan images,” *Chaos, Solitons & Fractals*, *140*, 110190, 2020.

[7] A. M. Ismael, and A. Şengür, “Deep learning approaches for COVID-19 detection based on chest X-ray images,” *Expert Systems with Applications*, *164*, 114054, 2020.

[8] G. Jain, D. Mittal, D. Thakur, and M. K. Mittal, “A deep learning approach to detect Covid-19 coronavirus with X-Ray images,” *Biocybernetics and Biomedical Engineering*, vol. 40, no. 4, pp. 1391-1405, 2020.

[9] C. Ouchicha, O. Ammor, and M. Meknassi, “CVDNet: A novel deep learning architecture for detection of coronavirus (Covid-19) from chest x-ray images,” *Chaos, Solitons & Fractals*, *140*, 110245, 2020.

[10] T. Ai, Z. Yang, H. Hou, C. Zhan, C. Chen, W. Lv, ... and L. Xia, “Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases,” *Radiology*, 200642, 2020.

[11] Y. Fang, H. Zhang, J. Xie, M. Lin, L. Ying, P. Pang, and W. Ji, “Sensitivity of chest CT for COVID-19: comparison to RT-PCR,” *Radiology*, 200432, 2020.

[12] A. A. Ardakani, A. R. Kanafi, U. R. Acharya, N. Khadem, and A. Mohammadi, “Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks,” *Computers in Biology and Medicine*, 103795, 2020.

[13] K. Zhang, X. Liu, J. Shen, Z. Li, Y. Sang, X. Wu, ... and L. Ye, “Clinically applicable AI system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography,” *Cell*, 2020.

[14] O. Attallah, D. A. Ragab, and M. Sharkas, “MULTI-DEEP: A novel CAD system for coronavirus (COVID-19) diagnosis from CT images using multiple convolution neural networks,” *PeerJ*, *8*, e10086, 2020.

[15] H. Yasar, and M. Ceylan, “A novel comparative study for detection of Covid-19 on CT lung images using texture analysis, machine learning, and deep learning methods,” *Multimedia Tools and Applications*, pp. 1-25, 2020.

[16] S. A. Harmon, T. H. Sanford, S. Xu, E. B. Turkbey, H. Roth, Z. Xu, ... and M. Blain, “Artificial intelligence for the detection of COVID-19 pneumonia on chest CT using multinational datasets,” *Nature communications*, vol. 11, no. 1, pp. 1-7, 2020.

[17] A. Jaiswal, N. Gianchandani, D. Singh, V. Kumar, and M. Kaur, “Classification of the COVID-19 infected patients using DenseNet201 based deep transfer learning,” *Journal of Biomolecular Structure and Dynamics*, pp. 1-8, 2020.

[18] Z. Li, Z. Zhong, Y. Li, T. Zhang, L. Gao, D. Jin, ... and J. Xiao, “From Community Acquired Pneumonia to COVID-19: A Deep Learning Based Method for Quantitative Analysis of COVID-19 on thick-section CT Scans,” *medRxiv*, 2020.

[19] A. M. Hasan, M. M. AL-Jawad, H. A. Jalab, H. Shaiba, R. W. Ibrahim, and A. A. R. AL-Shamasneh, “Classification of Covid-19 Coronavirus, Pneumonia and Healthy Lungs in CT Scans Using Q-Deformed Entropy and Deep Learning Features,” *Entropy*, vol. 22, no. 5, 517, 2020.

[20] H. Ko, H. Chung, W. S. Kang, K. W. Kim, Y. Shin, S. J. Kang, ... and J. Lee, “COVID-19 pneumonia diagnosis using a simple 2D deep learning framework with a single chest CT Image: Model Development and Validation,” *Journal of Medical Internet Research*, vol. 22, no. 6, e19569, 2020.

[21] S. Ahuja, B. K. Panigrahi, N. Dey, V. Rajinikanth, and T. K. Gandhi, “Deep transfer learning-based automated detection of COVID-19 from lung CT scan slices,” 2020.

[22] D. Dansana, R. Kumar, A. Bhattacharjee, D. J. Hemanth, D. Gupta, A. Khanna, and O. Castillo, “Early diagnosis of COVID-19-affected patients based on X-ray and computed tomography images using deep learning algorithm,” *Soft Computing*, pp. 1-9,2020.

[23] H. Panwar, P. K. Gupta, M. K. Siddiqui, R. Morales-Menendez, and V. Singh, “Application of Deep Learning for Fast Detection of COVID-19 in X-Rays using nCOVnet,” *Chaos, Solitons & Fractals*, 109944, 2020.

[24] M. E. Karar, E. E. D. Hemdan, and M. A. Shouman, “Cascaded deep learning classifiers for computer-aided diagnosis of COVID-19 and pneumonia diseases in X-ray scans,” *Complex & Intelligent Systems*, pp. 1-13, 2020.

[25] M. M. Rahaman, C. Li, Y. Yao, F. Kulwa, M. A. Rahman, Q. Wang, ... and X. Zhao, “Identification of COVID-19 samples from chest X-Ray images using deep learning: A comparison of transfer learning approaches,” *Journal of X-ray Science and Technology*, (Preprint), pp. 1-19, 2020.

[26] K. Elasnaoui, and Y. Chawki, “Using X-ray images and deep learning for automated detection of coronavirus disease,” *Journal of Biomolecular Structure and Dynamics*, (just-accepted), pp. 1-22, 2020.

[27] Anonim, “China National Center for Bioinformation 2019 Novel Coronavirus Resource (2019nCoVR)”, [Çevrimiçi]. Available at: http://ncov-ai.big.ac.cn/download?lang=en, (accessed 12 October 2020).

[28] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, “Dual path networks,” In *Advances in neural information processing systems*, pp. 4467-4475, 2017.

[29] K. K. Singh, M. Siddhartha, and A. Singh, “Diagnosis of Coronavirus Disease (COVID-19) from Chest X-ray images using modified XceptionNet,” *Romanian Journal of Information Science and Technology*, *23*, pp. 91-105, 2020.

[30] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-ResNet and the impact of residual connections on learning,” 31st AAAI. In *Conf Artif Intell AAAI 2017, 2017.*

[31] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, ... and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[32] C. Szegedy, S. Ioffe, and V. Vanhoucke, AA. et Alemi, “In Inception-v4, inception-resnet and the impact of residual connections on learning,” In Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, ... and A. Rabinovich, “Going deeper with convolutions,” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9, 2015.

[34] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818-2826, 2016.

[35] K. Simonyan, and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556,* 2014.

[36] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size,” *arXiv preprint arXiv:1602.07360,* 2016.

[37] E. Başaran, Z. Cömert, and Y. Çelik, “Convolutional neural network approach for automatic tympanic membrane detection and classification,” *Biomedical Signal Processing and Control*, *56*, 101734, 2020.

[38] C. Xu, J. Yang, H. Lai, J. Gao, L. Shen, and S. Yan, “UP-CNN: Un-pooling augmented convolutional neural network,” *Pattern Recognition Letters*, *119*, pp. 34-40, 2019.

[39] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” In *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 770-778, 2016.

[40] F. Chollet, “Keras”, 2015.

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:69-78 (2021),
Received: June 25, 2021 Accepted: August 30, 2021
JSCAI
ISSN: 2717-8226

Training Feedforward Neural Networks to Predict the Size of the Population by Using a New Hybrid Method Hestenes-Stiefel (HS) and Dai-Yuan (DY)

[1]Hisham M. Khudhur , [2]Khalil K. Abbo , [3]Aydin M. Khudhur

[1]Mathematics Department, College of Computer Science and mathematics, University of Mosul, Mosul, Iraq
[2]Department of Mathematics, College of Basic Education, University of Telafer, Tall'Afar, Iraq
[3]Department of Studies and planning, Presidency of telafer university, University of Telafer, Tall'Afar, Iraq

Corresponding author: First A. Author (e-mail:- hisham892020@uomosul.edu.iq).

ABSTRACT We proposed a new conjugate gradient type hybrid approach in this study, which is based on merging Hestenes-Stiefel and Dai-Yuan algorithms using the spectral direction conjugate algorithm, we showed their absolute convergence. Under some assumptions and they satisfied the gradient property. The numerical results demonstrate the efficacy of the developed feedforward neural network training approach. To estimate the size of the population using the Thomas Malthus population model, and Our numerical results were very close to the model of the Tomas Malthose Model, we can use the method to predict other problems through the use of ann.

KEYWORDS: Algorithms; ANN; Conjugate Gradient; Hybrid; Population.

1. INTRODUCTION

Multilayer feedforward neural networks (MLFFNN) are parallel computational models made up of densely interconnected, adaptable processing units that have an innate proclivity for learning from experience as well as discovering new information. They have been effectively employed in various domains of artificial intelligence [1],[2], [7], and [8] are often found to be more efficient and accurate than other classification techniques [17] due to their exceptional capability of self-learning and self-adapting. Feedforward neural networks (FNN) are often operated using the following equations:

$$net_j^i = \sum_{i=1}^{N_{L-1}} w_{i,j}^{i-1} x_j^{i-1} + b_j^i, O_j^l = f\left(net_j^l\right) \tag{1}$$

Where $f\ (net_j^l)$ is the activation function , net_j^l is the sum of the weight inputs for the j-th node in the l - th layer (j=1,2,…, Nl), $w_{i,j}$ is the weights from the i-th neuron to the j-th neuron at the $l-1, l-th$ layer ,respectively, b_j^l is the bias of the $j-th$ neuron at the $l-th$ layer and x_j^l is the output of the j-th neuron which belongs to the $l-th$ layer. The goal of training a neural network is to iteratively change its weights to minimize the difference between the network's actual output and the training set's desired output [26]. Finding such a minimum is actually the same as finding an optimal minimization of the error function, which is defined as:

$$E(w) = \frac{1}{2}\sum_{j=1}^{P} \sum_{i=1}^{M} \left(O_i^{(j)} - T_i^{(j)}\right)^2 \qquad (2)$$

The variables O_i and T_i are the desired and the actual output of the $i-th$ neuron, respectively. The index j denotes the particular learning pattern. The vector w is composed of all weights in the net. The most extensively used approach for training multilayer feedforward neural networks is backpropagation (BP). The weight vector w is adjusted using the steepest descent with respect to E in the typical backpropagation algorithm:

$$w_{k+1} = w_k - \alpha_k g_k, g_k = \nabla E(w_k) \qquad (3)$$

Where the constant α is the learning rate belongs to the interval (0,1) and w_k is a vector representing the weights at iteration (epoch) step k . The back propagation process takes an inordinate amount of time to modify the weights between the units in the network since the steepest descent method has a slow convergence rate and the search for the global minimum frequently becomes stranded at a bad local minimum[15]. As a result, many studies have proposed ways to improve this method, with several relying on a novel adaptive learning rate [4], [2], and [1]. Others utilize other cost functions or dynamic modification of the learning parameters [28], while others use the momentum term [27], [20], [13].Many people use weight initialization procedures that are unique to them [24]. The majority of them use higher order gradient optimization algorithms to reduce the appropriately error function [16], [22], a multivariable function that is dependent on the network's weights. However, the issue of speeding up the learning process remains. Especially when using big training sets and networks. The training of neural networks can be expressed as a non-linear unconstrained optimization problem [23], [3], [14].

The following is a breakdown of how this search is structured. The conjugate gradient algorithms are briefly described in Section 2. Section 3: A new hybrid conjugate gradient algorithm has been developed. Model of the Population, Section 4. Section 5 contains numerical comparisons and experiments.

2. CG TECHNIQUE

Due to their speed and simplicity, conjugate gradient (CG) methods are among the most often and efficiently utilized approaches for large-scale optimization issues. Due to their simplicity and minimal memory needs, conjugate gradient algorithms play a key role in rapidly training neural networks. They do not require the evaluation of the Hessian matrix or the impractical storage of an approximation of it. There are various conjugate gradient algorithms in the literature that have been extensively used for neural network training in a range of applications [5], [18]. The linear combination of the negative gradient vector at the current iteration with the previous search direction is the key idea for calculating the search direction. The method for determining the search direction is as follows:

$$d_1 = -g_1;\ d_{k+1} = -g_{k+1} + \beta_k d_k \tag{4}$$

Conjugate gradient methods differ in their way of defining the multiplier β_k . The most famous approaches were proposed by Fletcher Reeves (FR), Polak–Ribere (PR) and Hestenes–Stifel (HS) [11], [25], [12]:

$$\beta^{FR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}\ ,\ \beta^{PR} = \frac{g_{k+1}^T y_k}{g_k^T g_k}\ ,\ \beta^{PR} = \frac{g_{k+1}^T y_k}{g_k^T y_k}$$

The conjugate gradient methods using β^{FR} update were shown to be globally convergent [9], [21], [19]. However the corresponding methods using β^{PR} or β^{HS} update are generally more efficient ever without satisfying the global convergence property. [6] In the convergence analysis and implementations of CG methods, one often requires the inexact lien search such as the Wolfe line search. The standard Wolfe line search requites σ_k satisfying:

$$E(W_k + \alpha_k d_k) \le E(W_k) + \rho \alpha_k g_k^T d_k \tag{5}$$

$$g(W_k + \alpha_k d_k)^T d_k \ge \sigma g_k^T d_k \tag{6}$$

or strong Wolfe line search:

$$E(W_k + \alpha_k d_k) \le E(W_k) + \rho \alpha_k g_k^T d_k \tag{7}$$

$$\left| g_{k+1} + d_k \right| \le -\sigma g_k d_k \tag{8}$$

Where $0 < \rho < \sigma < 1$

Moreover, an important issue of CG algorithms is that when the search direction (4) fails to be descent (by Descent, we mean $g_k^T d_k < 0\ \forall k$ directions we restart the algorithm using the negative gradient direction to grantee convergence . A more sophisticated and popular restarting is the Powell restart.

$$\left| g_{k+1}^T g_k \right| \ge 0.2 \left\| g_{k+1} \right\|^2 \tag{9}$$

Where, $\| \ \|$ denotes to the Euclidean norm. Other important issue for then CG methods is that the search directions generated from equation (4) are conjugate if the objective function is convex and line search is exact i.e:

$$d_i^T G d_j = 0, \forall i \neq j \tag{10}$$

Where, G is the Hessian matrix for the objective function. Dai and Lioa in [10] showed that the equation (10) can be written as follows:

$$d_{k+1}^T y_k = 0 \tag{11}$$

which is called pure conjugacy condition and generalize to the

$$d_{k+1}^T y_k = t\, g_{k+1}^T s_k \quad , \quad t > 0 \quad , \quad s_k = W_{k+1} - W_k \tag{12}$$

for general objective function with inexact line search.

3. A NEW CONJUGATE ALGORITHM HYBRID ALGORITHM

We will derive New in this section. Unconstrained optimization using a hybrid conjugate gradient technique. Using the direction conjugate algorithm, the Hestenes-Stiefel and Dai-Yuan algorithms were combined [16]. The formula for determining direction is known to us

$$d_{k+1} = -g_{k+1} + \beta_{k+1} d_k \tag{13}$$

Hestenes-Stiefel algorithm

$$\beta_k^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k} \tag{14}$$

Dai-Yuan algorithm

$$\beta_k^{DY} = \frac{\|g_{k+1}\|^2}{y_k^T d_k} \tag{15}$$

Suppose that

$$\beta_{k+1}^{*} = \eta \beta_{k+1}^{HS} \tag{16}$$

$$\beta_{k+1}^{KH1} = \eta \beta_{k+1}^{DY} + (1-\eta)\beta_{k+1}^{HS} \tag{17}$$

$$d_{k+1} = -g_{k+1} + \eta \beta_{k+1}^{HS} d_k \tag{18}$$

$$d_{k+1} = -g_{k+1} + (\eta \beta_{k+1}^{DY} + (1-\eta)\beta_{k+1}^{HS}) d_k \tag{19}$$

Equality of equation (18) with (19) and note of the d_k equal in equations (18) and (19) we get

$$-g_{k+1} + \eta\beta_{k+1}^{HS} d_k = -g_{k+1} + (\eta\beta_{k+1}^{DY} + (1-\eta)\beta_{k+1}^{HS})d_k \tag{20}$$

Subtracting the g_{k+1} of two said from above equation we have

$$\eta\beta_{k+1}^{HS} d_k = (\eta\beta_{k+1}^{DY} + (1-\eta)\beta_{k+1}^{HS})d_k \tag{21}$$

After some algebra, we get

$$\eta = \frac{\beta_{k+1}^{HS}}{2\beta_{k+1}^{HS} - \beta_{k+1}^{DY}} \tag{22}$$

Substituting η in the equation (19)

$$\beta_{k+1}^{KH1} = \frac{\beta_{k+1}^{HS}\beta_{k+1}^{DY}}{2\beta_{k+1}^{HS} - \beta_{k+1}^{DY}} + (1 - \frac{\beta_{k+1}^{HS}}{2\beta_{k+1}^{HS} - \beta_{k+1}^{DY}})\beta_{k+1}^{HS} \tag{23}$$

After some algebra of above equation we get a new formula denote by β_{k+1}^{KH1} is defined by

$$\beta_{k+1}^{KH1} = \frac{(g_{k+1}^T y_k)^2}{d_k^T y_k (2g_{k+1}^T y_k - \|g_{k+1}\|^2)} \tag{24}$$

Substituting above equation in spectral direction conjugate algorithm, which is developed [1].

There for we have

$$d_{k+1} = -(\xi + \beta_{k+1}^{KH1} \frac{d_k^T y_k}{\|g_{k+1}\|^2}) g_{k+1} + \beta_{k+1}^{KH1} d_k \tag{25}$$

New Algorithm KH1

Step[1]:- Initialize w_1 and choose σ , ρ such that $0<\rho<\sigma<1,\ \xi \in [0,1]$ $E_G, \varepsilon > 0$ and $K_{\max}$, set $k = 1$.

Step[2]:- Calculate the error function value E_k and its gradient g_k .

Step[3]:- If $(E_k < E_G)$ or $\|g_k\| < \varepsilon$, set $w^* = w_k$ and $E^* = E_k$, return goal is meet and stop.

Step[4]:- compute the descent direction :

if $k = 1$ then, $d_k = -g_k$ go to step 6

Else

$$\beta_{k+1}^{KH1} = \frac{(g_{k+1}^T y_k)^2}{d_k^T y_k (2g_{k+1}^T y_k - \|g_{k+1}\|^2)} \text{ and then compute:}$$

$$d_{k+1} = -(\xi + \beta_{k+1}^{KH1} \frac{d_k^T y_k}{\|g_{k+1}\|^2}) g_{k+1} + \beta_{k+1}^{KH1} d_k \,.$$

Step[5]:- Compute the learning rate α_k by line search procedure, such the standard Wolfe conditions (5) and (6).

Step[6]:- update the weights:

$w_{k+1} = w_k + \alpha_k d_k$ and set $k = k + 1$.

Step[7]:- If $k > k_{\max}$ return Error goal not meet and stop else go to step (2).

4. POPULATION MODEL

The term "population" refers to all living organisms of the same type that reside in the same geographical area. The phrase "population" is used at the conference Sociology aware to describe to the human people who dwell in a country or region group. And the special science of demography is concerned with the statistical elements of human population.

The topic of modeling the population of the key issues that affected the environment Ecology science is the application of mathematical modeling for the study of movement) dynamic organisms in the growth and decay. Changes in population sizes as a result of interactions between individuals in the natural environment with members of the same gender, as well as other types of living animals, are included in the population modeling study. Knowing the population is one of the most important concerns for various countries around the world, and many conduct censuses every ten years to determine the true population numbers, and the importance of knowing the population in its close association with various aspects of life in human societies, and strong development plans and their relationship to and provision of [16].

It was true for population modeling beginning in the eighteenth century with the development of tools for modeling the change in the number of individuals to comprehend population expansion and contraction. When contemplating the fate of humanity, British scholar Thomas Malthus Thomas Robert Malthus (1834-1766) and one of the pioneers in this field, to remark that the population of human beings grows according to a geometric pattern [16]. Thomas Malthus' formula for population growth

$N(t) = N(0)e^{ct}$

t	The time
$N(t)$	Number of people
$N(0)$	Number of people in time $t = 0$

$$N(t) = N(0)e^{0.3t}; N(0) = 3.9$$

5. EXPERIMENTS AND RESULTS

The performance of the algorithm KH1 has been studied using a computer simulation. The simulations were run in MATLAB (7.6) win8 and hp laptop, and the MSBP's performance was evaluated and compared to batch versions of the above approach. and In this part of the research we write the results of the learned artificial neural network and a comparison with the results of the Thomas Malthus Modeling, and as shown in table (1) and drawing (1) as drawing (1) illustrates the comparison between the Modeling results and the artificial neural network results, and the drawing (2) shows the square error rate resulting from training the artificial neural network.

Table 1. ANN Solution vs. Thomas Malthus Modeling

Thomas Malthus Modeling Population (million people)	ANN Population (million people)
4.1194	4.1157
4.1437	4.1506
4.1681	4.1739
4.1927	4.1903
4.2673	4.265
4.3689	4.3682
4.473	4.4666
4.6065	4.6152
4.6337	4.6402
4.661	4.6709
4.7161	4.7222
4.744	4.7374
4.8569	4.8474
5.0019	5.0044
5.0314	5.0257
5.0611	5.0559
5.0909	5.0917
5.121	5.1236
5.1512	5.146
5.5279	5.5211
5.7265	5.7181
6.4037	6.407
6.7122	6.7165
6.8721	6.8684
7.0772	7.0869

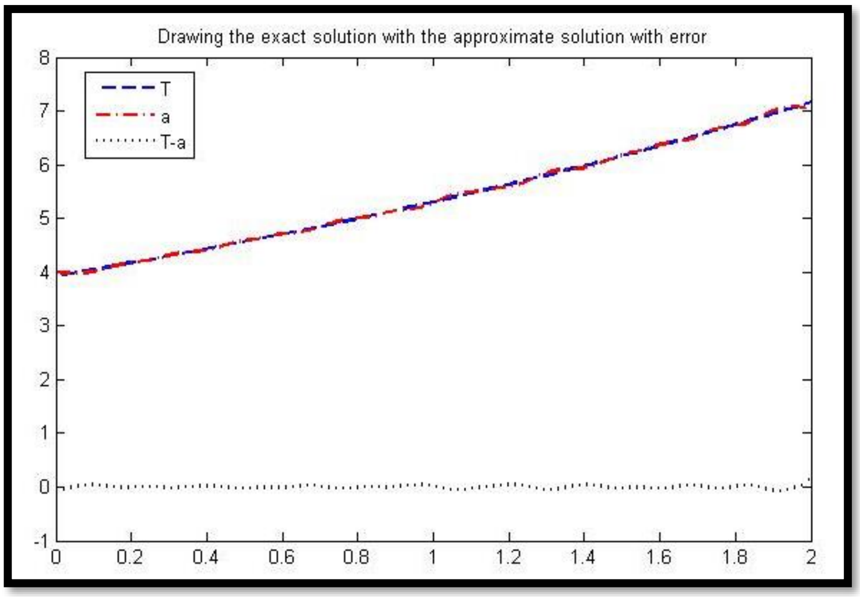

Figure.1. Comparison between the Modeling results and the artificial neural network results with error

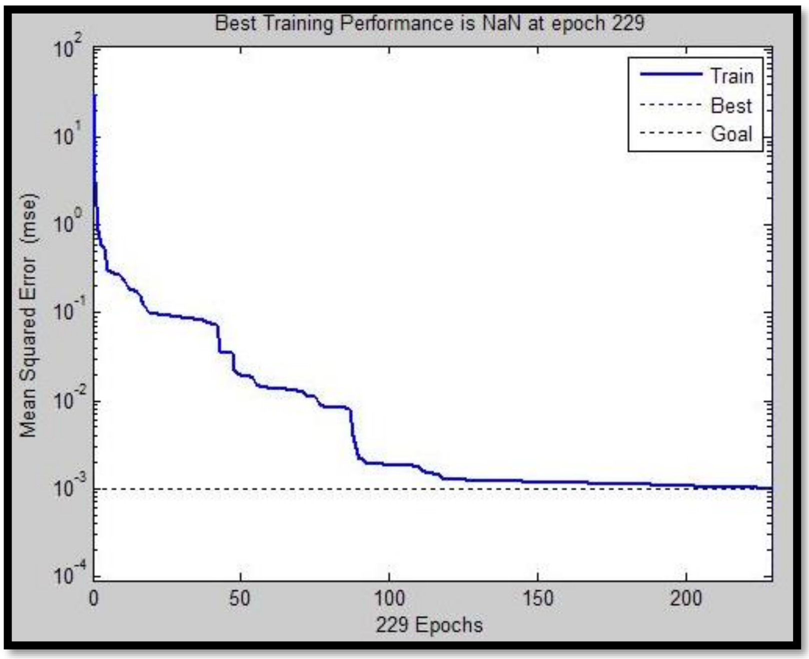

Figure.2. The Mean Squared Error (MSE)

6. CONCLUSIONS

In this work, we have proposed a new CG method for training neural networks which are based on the hybrid algorithm Hestenes-Stiefel and Dai-Yuan. The proposed method preserves the strong convergence properties and descent property. The proposed method is suitable for training large-scale neural networks. Our numerical experiments have shown that the proposed method efficient to predict the size of the population, We can use the method to predict other problems through the use of neural networks, as well as solving optimization, fuzzy optimization problems, solving fuzzy equations, and fuzzy neural networks.

REFFRNCES

[1] Abbo, K. and Hind, M. 2012. Improving the learning rate of the Backpropagation Algorithm Aitken process'. *Iraqi Journal of the statistical sciences, accepted (to appear)*. (2012).

[2] Abbo, K. and Mohammed, H. 2014. Conjugate Gradient Algorithm Based on Aitken's Process for Training Neural Networks. *AL-Rafidain Journal of Computer Sciences and Mathematics*. 11, 1 (2014). DOI:https://doi.org/10.33899/csmj.2014.163730.

[3] Abbo, K.K. and Khudhur, H.M. 2016. New A hybrid conjugate gradient Fletcher-Reeves and Polak-Ribiere algorithm for unconstrained optimization. *Tikrit Journal of Pure Science*. 21, 1 (2016), 124–129.

[4] Abbo, K.K. and Khudhur, H.M. 2016. New A hybrid Hestenes-Stiefel and Dai-Yuan conjugate gradient algorithms for unconstrained optimization. *Tikrit Journal of Pure Science*. 21, 1 (2016), 118–123.

[5] Abbo, K.K., Laylani, Y.A. and Khudhur, H.M. 2016. Proposed new Scaled conjugate gradient algorithm for Unconstrained Optimization. *International Journal of Enhanced Research in Science, Technology & Engineering*. 5, 7 (2016).

[6] ABBO, K.K., Laylani, Y.A. and Khudhur, H.M. 2018. A NEW SPECTRAL CONJUGATE GRADIENT ALGORITHM FOR UNCONSTRAINED OPTIMIZATION. *International Journal of Mathematics and Computer Applications Research (IJMCAR)*. 8, (2018), 1–9.

[7] Abdullah, Z.M., Hameed, M., Hisham, M.K. and Khaleel, M.A. 2019. Modified new conjugate gradient method for Unconstrained Optimization. *Tikrit Journal of Pure Science*. 24, 5 (2019), 86–90.

[8] Ahmed, A.S. 2018. *Optimization Methods For Learning Artificial Neural Networks*. University of Mosul.

[9] Al-Baali, M. 1985. Descent property and global convergence of the Fletcher—Reeves method with inexact line search. *IMA Journal of Numerical Analysis*. 5, 1 (1985), 121–124.

[10] Dai, Y.H. and Liao, L.Z. 2001. New conjugacy conditions and related nonlinear conjugate gradient methods. *Applied Mathematics and Optimization*. 43, 1 (2001), 87–101. DOI:https://doi.org/10.1007/s002450010019.

[11] Fletcher, R. and Reeves, C.M. 1964. Function minimization by conjugate gradients. *The Computer Journal*. 7, 2 (1964), 149–154. DOI:https://doi.org/10.1093/comjnl/7.2.149.

[12] Hestenes, M.R. and Stiefel, E. 1952. *Methods of conjugate gradients for solving linear systems*. NBS Washington, DC.

[13] Hmich, A., Badri, A. and Sahel, A. 2011. Automatic speaker identification by using the neural network. *International Conference on Multimedia Computing and Systems -Proceedings* (2011).

[14] Jabbar, H.N., Abbo, K.K. and Khudhur, H.M. 2018. Four--Term Conjugate Gradient (CG) Method Based on Pure Conjugacy Condition for Unconstrained Optimization. *kirkuk university journal for scientific studies*. 13, 2 (2018), 101–113.

[15] Khudhur, H.M. 2020. Modified Barzilai-Borwein Method for Steepest Descent Method to Solving Fuzzy Optimization Problems(FOP). *Albahir journal*. 12, 23–24 (2020), 63–72.

[16] Khudhur, H.M. 2015. *Numerical and analytical study of some descent algorithms to solve unconstrained Optimization problems*. University of Mosul.

[17] Khudhur, H.M. and Abbo, K.K. 2021. A New Conjugate Gradient Method for Learning Fuzzy Neural Networks. *Journal of Multidisciplinary Modeling and Optimization*. 3, 2 (2021), 57–69.

[18] Khudhur, H.M. and Abbo, K.K. 2021. A New Type of Conjugate Gradient Technique for Solving Fuzzy Nonlinear Algebraic Equations. *Journal of Physics: Conference Series*. 1879, 2 (2021), 22111. DOI:https://doi.org/10.1088/1742-6596/1879/2/022111.

[19] Khudhur, H.M. and Abbo, K.K. 2021. New hybrid of Conjugate Gradient Technique for Solving Fuzzy Nonlinear Equations. *Journal of Soft Computing and Artificial Intelligence*. 2, 1 (2021), 1–8.

[20] Lange, N., Bishop, C.M. and Ripley, B.D. 1997. Neural Networks for Pattern Recognition. *Journal of the American Statistical Association*. 92, 440 (1997). DOI:https://doi.org/10.2307/2965437.

[21] Laylani, Y.A., Abbo, K.K. and Khudhur, H.M. 2018. Training feed forward neural network with modified Fletcher-Reeves method. *Journal of Multidisciplinary Modeling and Optimization*. 1, 1 (2018), 14–22.

[22] Livieris, I.E. and Pintelas, P. 2012. An Advanced Conjugate Gradient Training Algorithm Based on a Modified Secant Equation. *ISRN Artificial Intelligence*. 2012, (2012). DOI:https://doi.org/10.5402/2012/486361.

[23] Livieris, I.E., Sotiropoulos, D.G. and Pintelas, P. 2009. On descent spectral CG algorithms for training recurrent neural networks. *PCI 2009 - 13th Panhellenic Conference on Informatics* (2009).

[24] Nguyen, D. and Widrow, B. 1990. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. *1990 IJCNN International Joint Conference on Neural Networks* (1990), 21–26.

[25] Polak, E. and Ribiere, G. 1969. Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*. 3, R1 (1969), 35–43.

[26] Rumelhart, D.E., Hinton, G.E. and Williams, R.J. 2013. Learning Internal Representations by Error Propagation. *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*.

[27] Svozil, D., Kvasnička, V. and Pospíchal, J. 1997. Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems* (1997).

[28] Walczak, S. and Cerpa, N. 1999. Heuristic principles for the design of artificial neural networks. *Information and Software Technology*. 41, 2 (1999). DOI:https://doi.org/10.1016/S0950-5849(98)00116-5.

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:79-84 (2021)
Received: June 19, 2021 Accepted: August 30, 2021
JSCAI
ISSN: 2717-8226

Usage Areas and Thermal Performance of Nanofluids and Nanoparticles

[1]Edip Taşkesen , [2]Khandan Roshanaei ,[3] Mehmet ÖZKAYMAK

[1,2,3] Energy Systems Engineering Department, Karabuk University, Karabuk, Turkey

Corresponding author: First A. Author (e-mail: ediptaskesen@karabuk.edu.tr).

ABSTRACT In the study, it has been observed that there are many alternatives for the usage areas of nanofluids formed by dispersing solid particles of nanometric size (1-100 nm) in a basic fluid, as well as these fluids are efficient in both solar energy systems and other thermal systems. In this study, widely used nanofluids in heating and cooling systems and their application areas were investigated. It was observed that when nanofluids with different parameters are used, it affects thermal conductivity efficiency.

KEYWORDS: Nanofluids, Heat Transfer, Nanoparticles

1. INTRODUCTION

Improving thermal conductivity efficiency is one of the ways to use energy efficiently. Nanofluids, with their improved thermo physical properties and heat convey performances with contrast to conventional substances heat transfer; attract the attention of researchers to be used in many different heat transfer applications. Nanofluids have several techniques used to boost and improve and speed up the heat transfer. The heat transfer rate changeable to flow in geometry, boundary conditions, or improving thermo-physical properties. nanofluids usually have a median size<100 nm and also various dimensions, formations and structure [1]. As seen in Fig1 and Fig2, nanofluid basic fluid consists of a nanoparticle and a nanolayer [2].

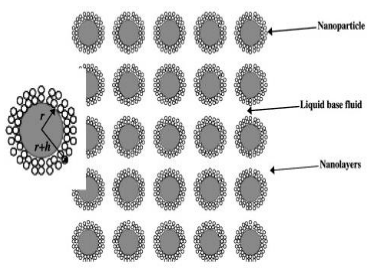

Figure 1. Nanofluid schematic view [2].

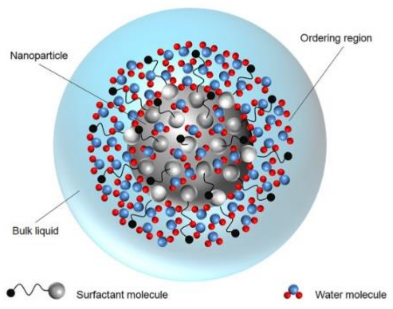

Figure 2. Nanoparticle-fluid structure [3].

Nanofluids have boosted heat transport shifting and better productivity as a higher energy in a variety of thermal altercation systems for mentioned industrial demands; shipment, electronic cooling, energy storage, applications etc. Nanofluids can be applied in various application and s such as commercial cooling, heating constructions, decreasing contamination, nuclear applications cooling, space engineering, friction depletion, magnetic sealing, antibacterial undertaking, nano drug systems, biological fuel cells [4,5]. Various nano particle materials are used to obtain nanofluids, for example, metals (Al, Cu, Ag), metal oxide (Al_2O_3, CuO, MgO), Nitrides (AlN, SiN), carbide ceramics (SiC, TiC), thermal resistors and semiconductor units (TiO_2, SiO_2) [6].

As can be noticed and visible in Fig3., the heat conduction of nanofluids varies depending on the nanoparticle concentration, size, shape, thermal conductivity, basic fluid type, nanofluid temperature and preparation technique, which are different parameters [7].

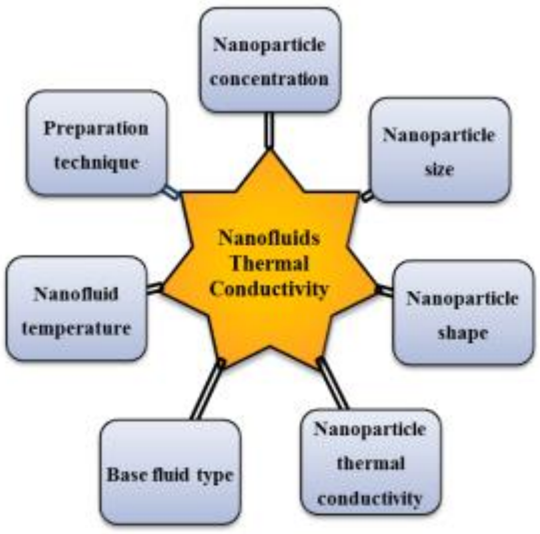

Figure 3. The reaction of different parameters on the heated dynamism of nanofluids [7].

In this study; nanofluids, their application areas and elements influencing the change of heated conductivity were investigated.

2. MATERIALS METHOD

Recently, there has been an increasing interest in studies on the heated conductivity of nanofluids. The first experimental study on nanofluids was carried out by Choi and in this study, by adding nanoparticles into the basic fluid, the thermal conductivity and heat transfer performance of the fluid was investigated [8]. Heris has experimentally investigated the change of boiling heat transfer in automotive cooling systems using an ethylene-glycol-water based (60%: 40%) nanofluid consisting of 0.5% and 40 nm sizes CuO nanoparticles. He stated that there is a 55% increase in heat transfer compared to the basic fluid [9]. Abbassi et al. Analytically considered the heat removal of 10 nm TiO2/water nanofluid at different volumetric concentrations (0.25%, 0.5%, 1.0% and 1.5%) at different Reynolds numbers. It has been found that the heat transfer coefficient of nanofluid higher than the basic fluid water [10]. Ali et al. Conducted an experimental study using ZnO/water nanofluid at different volumetric concentrations (0.01%, 0.08%, 0.2% and 0.3%) to improve the heat transfer performance of a car radiator. They observed a 46% increase in heat transfer at 0.2% volumetric concentration compared to the basic fluid, water [11]. In their work, Li et al. Created an experimental system to investigate the convective heat transfer and flow properties of nanofluid in a tube. They measured both the convective heat transfer coefficient and the friction factor of the Cu/water nanofluid for laminar and turbulent flow. They discussed in detail the effects of factors such as volume fraction and Reynolds numbers of suspended nanoparticles on heat transfer and flow properties. Compared to the base fluid, for example, with amount of 2.0% Cu as nanoparticles in volumetric concentration at the identical Reynolds number, they observed that the coefficient of convective heat transfer for the nanotechnology increased by approximately 60%. Taking the considering the elements altering the convective heat transfer coefficient of nanofluid, they established a new convective heat transfer interaction for nanofluid under mono-phase motion in ducts and pipelines. In the contrast of the middle of the experimental data and the deliberated outcomes, they showed that the association accurately describes the energy transport of the nanofluid [12]. Tekir et al., examine experimentally the forced convection heat transfer of Fe_3O_4/water nanofluid flow at different volumetric concentrations ($0 \leq \phi \leq 0.05$) in laminar flow ($1122 < Re < 2124$) in a straight pipe under the influence of a constant magnetic field (B= 0.3T). Experimental results have shown that the fixed magnetic field provides 13% increase in convective heat transfer compared to the absence of the magnetic field [13]. Baskar et al experimentally investigated the heat transfer by using MWCNT/water-Ethylene Glycol (EG) nanofluid in a 2500 mm long, 10.7 mm inner diameter and 12.7 mm outer diameter copper pipe. The increase in the heat transfer coefficient was found respectively 30% and 34.74%, and in volumetric concentrations of MWCNT at 0.15% and 0.3%. However, for the same nanofluids, they observed that the heat transfer coefficient increased significantly as flow conditions changed (from laminar to turbulent flow) [14].

As shown in Fig4. And Table1. it is seen that the ones with the highest thermal conductivity are inorganic metals (Melamine-formaldehyde), basic fluid (water), metals (Cu) and metal oxides (ZnO) [15].

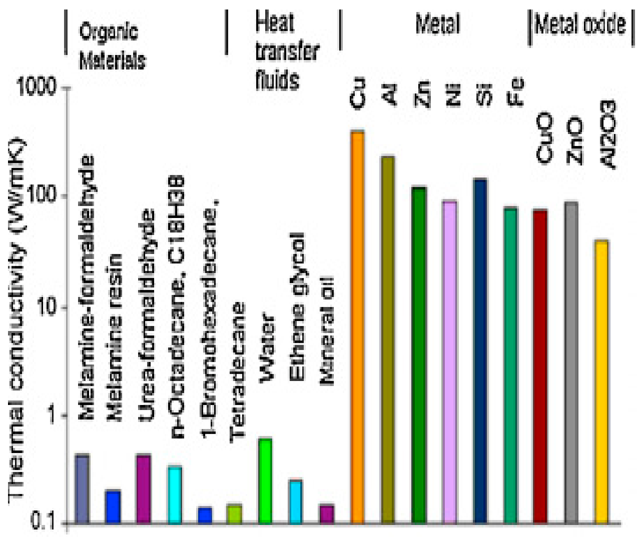

Figure 4. Thermal conductivity comparison [15].

Table 1. Thermal conductivity of nanoparticles [16].

Sl.no	Nanopowders	Thermal conductivity (W/m° K)
1	Aluminum Oxide (Al_2O_3)	40
2	Zinc Oxide (ZnO)	29
3	Tin Oxide (SnO_2)	36
4	Iron Oxide (Fe_2O_3)	7
5	Gold nanopowder (Au)	315
6	Titanium Dioxide (TiO_2)	8.5
7	Copper Oxide (CuO)	76
8	Carbon nanotubes	3000–6000
9	Zirconium (IV) Oxide (ZrO_2)	2
10	Silicon nitride (Si_3N_4)	29–30
11	Boron nitride (BN)	30–33
12	Aluminum nitride (AlN)	140–180
13	Diamond nanopowder (C)	900
14	Silver nanopowder (Ag)	424

As can be seen in Fig. 6 and Fig. 7, various application areas of nanofluids are emphasized to use energy more efficiently.

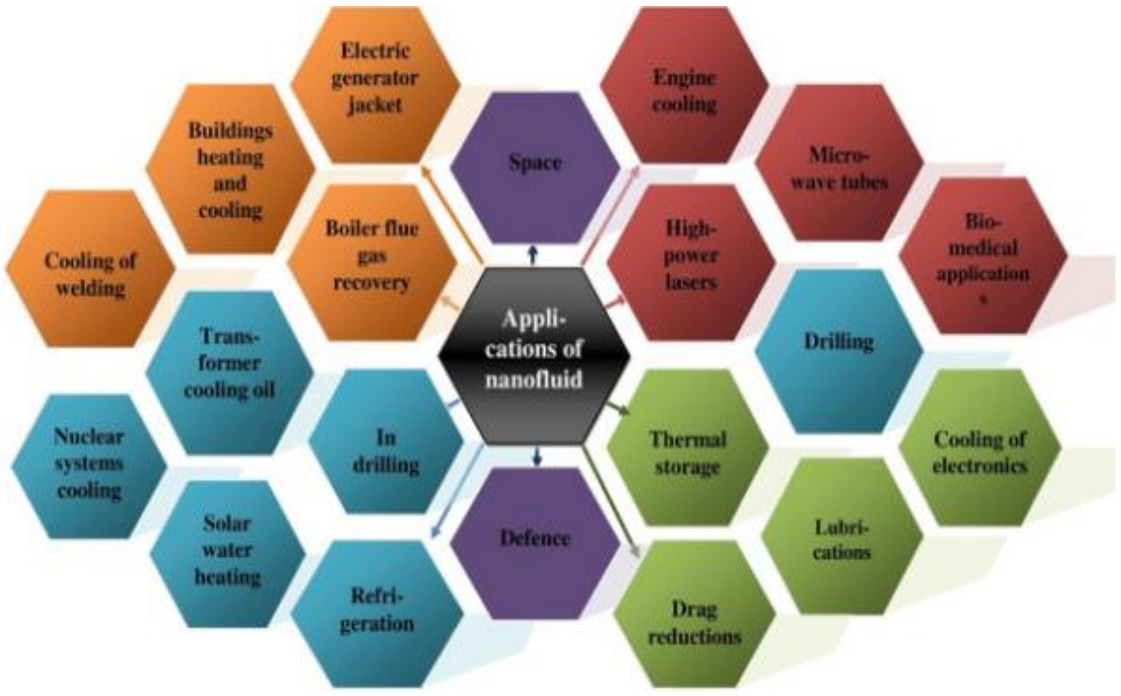

Figure 5. Application areas of nanofluid [6].

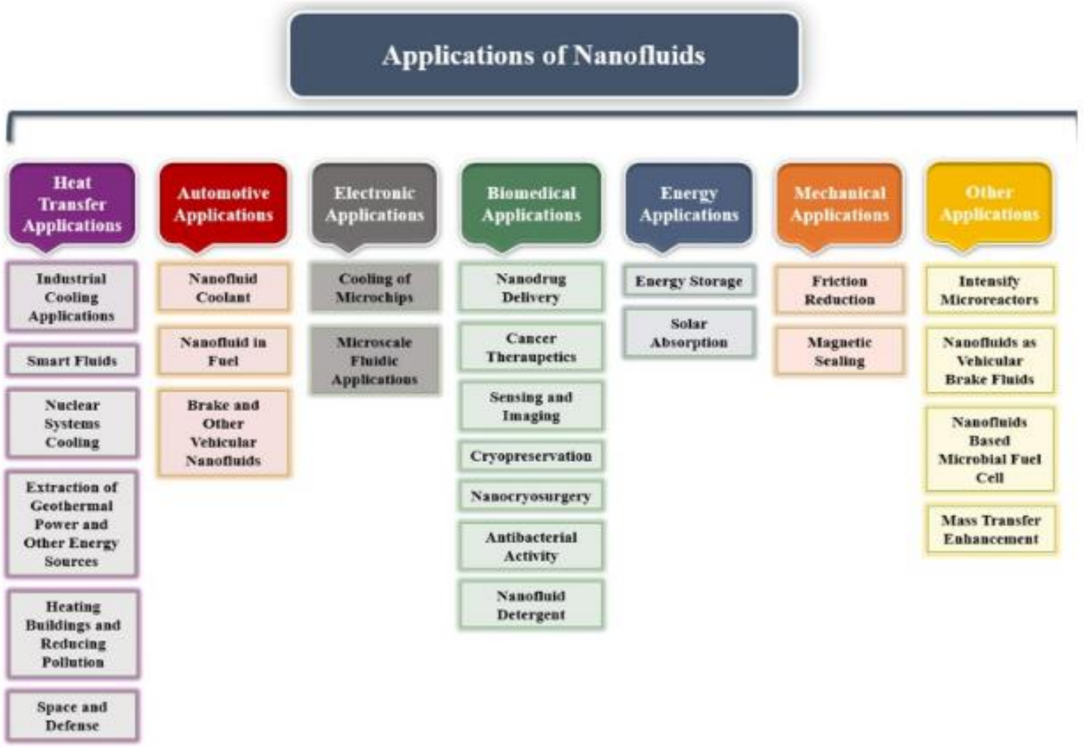

Figure 6. Implementation of nanofluids [17].

3. CONCLUSION

In this study; Research was done on nanofluids and their application areas. More efficiency is obtained in heating and cooling systems by using nanofluids. It has been observed that the thermal efficiency of nanofluids varies depending on different parameters (nanoparticle concentration, size, shape, thermal conductivity, bases of fluid type, nanofluid temperature, and preparation technique) and the nanoparticle with the highest thermal conductivity is the Carbon nanotube.

REFERENCES

[1] Colla, L., Fedele, L., Manca, O., Marinelli, L., and Nardini, S., ‘Experimental and numerical investigation on forced convection in circular tubes with nanofluids’, ***Heat Transfer Engineering***, 37 (13–14): 1201–1210 (2016).

[2] Yu, W. and Choi, S. U. S., ‘The role of interfacial layers in the enhanced thermal conductivity of nanofluids: a renovated Maxwell model’, ***Journal Of Nanoparticle Research***, 5 (1): 167–171 (2003).

[3] Arshad, A., Jabbal, M., Yan, Y., and Reay, D., ‘A review on graphene based nanofluids: Preparation, characterization and applications’, ***Journal Of Molecular Liquids***, 279: 444–484 (2019).

[4] Devendiran, D. K. and Amirtham, V. A., ‘A review on preparation, characterization, properties and applications of nanofluids’, ***Renewable And Sustainable Energy Reviews***, 60: 21–40 (2016).

[5] Bashirnezhad, K., Bazri, S., Safaei, M. R., Goodarzi, M., Dahari, M., Mahian, O., Dalkılıça, A. S., and Wongwises, S., ‘Viscosity of nanofluids: a review of recent experimental studies’, ***International Communications In Heat And Mass Transfer***, 73: 114–123 (2016).

[6] Aglawe, K. R., Yadav, R. K., and Thool, S. B., ‘Preparation, applications and challenges of nanofluids in electronic cooling: A systematic review’, ***Materials Today: Proceedings***, (2021).

[7] Tawfik, M. M., ‘Experimental studies of nanofluid thermal conductivity enhancement and applications: A review’, ***Renewable And Sustainable Energy Reviews***, 75 (November 2016): 1239–1253 (2017).

[8] Choi, S. S. and Eastman, A. A., ‘Enhancing thermal conductiivity of fluids with nanoparticles’, (1995).

[9] Heris, S. Z., Etemad, S. G., and Esfahany, M. N., ‘Experimental investigation of oxide nanofluids laminar flow convective heat transfer’, ***International Communications In Heat And Mass Transfer***, 33 (4): 529–535 (2006).

[10] Abbassi, Y., Talebi, M., Shirani, A. S., and Khorsandi, J., ‘Experimental investigation of TiO2/Water nanofluid effects on heat transfer characteristics of a vertical annulus with non-uniform heat flux in non-radiation environment’, ***Annals Of Nuclear Energy***, 69: 7–13 (2014).

[11] Ali, H. M., Ali, H., Liaquat, H., Maqsood, H. T. Bin, and Nadir, M. A., ‘Experimental investigation of convective heat transfer augmentation for car radiator using ZnO–water nanofluids’, ***Energy***, 84: 317–324 (2015).

[12] Li, Q. and Xuan, Y.-M., ‘Flow and Heant Transfer Performances of Nanofluids Inside Small Hydraulic Diameter Flat Tube’, ***Journal Of Engineering Thermophysics***, 25 (2): 305–307 (2004).

[13] Tekir, M., Taskesen, E., Aksu, B., Gedik, E., and Arslan, K., ‘Comparison of bi-directional multi-wave alternating magnetic field effect on ferromagnetic nanofluid flow in a circular pipe under laminar flow conditions’, ***Applied Thermal Engineering***, 179: 115624 (2020).

[14] Baskar, S., Chandrasekaran, M., Vinod Kumar, T., Vivek, P., and Karikalan, L., ‘Experimental studies on convective heat transfer coefficient of water/ethylene glycol-carbon nanotube nanofluids’, ***International Journal Of Ambient Energy***, 41 (3): 296–299 (2020).

[15] Saidur, R., Leong, K. Y., and Mohammed, H. A., ‘A review on applications and challenges of nanofluids’, ***Renewable And Sustainable Energy Reviews***, 15 (3): 1646–1668 (2011).

[16] Elango, T., Kannan, A., and Murugavel, K. K., ‘Performance study on single basin single slope solar still with different water nanofluids’, ***Desalination***, 360: 45–51 (2015).

[17] Olfian, H., Ajarostaghi, S. S. M., and Ebrahimnataj, M., ‘Development on evacuated tube solar collectors: A review of the last decade results of using nanofluids’, ***Solar Energy***, 211: 265–282 (2020).

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:85-97 (2021)
Received: October 22, 2021 Accepted: November 15, 2021
JSCAI
ISSN: 2717-8226

Application of Social Spider Optimization for Permutation Flow Shop Scheduling Problem

Mohamed Kurdi

Computer Engineering Department, Aydın Adnan Menderes University, Turkey
Corresponding author: First A. Author (e-mail: mohamed.kurdi@adu.edu.tr).

ABSTRACT Permutation flow shop scheduling problem (PFSP) is an NP-complete problem with a wide range of applications in many real-world applications. Social spider optimization (SSO) is a swarm intelligence algorithm proposed for continuous optimization problems. Recently, SSO has received increased interest in the field of combinatorial optimization as well. For this reason, in this paper, SSO algorithm is proposed to solve the PFSP with make span minimization. The proposed algorithm has been tested on 141 well-known benchmark instances and compared against six other conventional and best-so-far metaheuristics. The obtained results show that SSO outperforms some of the compared works although they are hybrid methods.

KEYWORDS: Metaheuristic, Optimization, Flow Shop Scheduling, Social Spider, Swarm Intelligence.

1. INTRODUCTION

Flow shop is a renowned manufacturing layout in which a set of jobs should be processed, in the same order, on a set of machines. The flow shop scheduling problem considers the sequence of the jobs over the machines with respect to a certain performance measure, such as makespan, maximum lateness, or total weighted completion time. If each machine should process the jobs in the same order, the problem is called as permutation flow shop scheduling problem (PFSP) [1]. PFSP is an NP-complete problem that has several real-life application fields, such as computing designs, production, information processing, communications, and transportation [2].

Due to its complexity and practical relevance, PFSP has been addressed by a considerable number of metaheuristic algorithms. These algorithms include discrete jaya algorithm [3], genetic-shuffled frog-leaping [4], Iterative beam search [5], evolutionary algorithm [6], hybrid backtracking search [2], shuffled complex evolution [7], genetic algorithm [8-10] bacterial foraging optimization [11], bat algorithm (BA) [12], rhinoceros search [13], biogeography-based optimization [14], differential evolution [15] [16], harmony search [17], cuckoo search [18], scatter search [19], iterated greedy [20], monkey search [21], and ant colony optimization (ACO) [22].

Social spider optimization (SSO) is a new swarm intelligence algorithm that has been proposed for continuous optimization problems [23]. However, recently, there has been an increased interest in applying it for solving combinatorial problems. Works such as [24-28] have shown it as a promising area of research for combinatorial problems.

In this paper, SSO is proposed for the PFSP with makespan minimization. The aim is to examine the effectiveness of SSO on PFSP as it has been widely used as a benchmark problem to validate the effectiveness of many optimization algorithms. To the best of our knowledge, there is no published work to address the PFSP by using this algorithm. The remainder of this paper is structured as follows. The problem definition is given in the next section. Section 3 describes the proposed algorithm. The computational results are reported in Section 4. The conclusion and future works are presented in Section 5.

2. PROBLEM DEFINITION

Suppose that *n* jobs $\{J_i\}_{i=1}^n$ need to be sequentially processed on *m* machines $\{M_k\}_{k=1}^m$. Each job J_i is composed of *m* operations $(O_{i1}, O_{i2}, \ldots, O_{im})$. All jobs should have the same processing order on each machine. O_{ik} represents the operation of job J_i on machine M_k which needs using M_k solely for a specified continued time called P_{ik} (pre-emption is not allowed). Let $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$ be a permutation of the jobs in which π_i denotes the index of the job placed at the *i*th position of *π*. Then, the completion time of each job $C(\pi_i, k)$, $i = 1, \ldots, n$ can be calculated by the following set of recursive formulas [29].

$$C(\pi_1, 1) = p_{\pi_1 1}, \tag{1}$$

$$C(\pi_i, 1) = C(\pi_{i-1}, 1) + p_{\pi_i 1}, \quad i = 2, \ldots, n, \tag{2}$$

$$C(\pi_1, k) = C(\pi_1, k-1) + p_{\pi_1 k}, \quad k = 2, \ldots, \mathrm{m}, \tag{3}$$

$$C(\pi_i, k) = \max\{C(\pi_{i-1}, k), C(\pi_i, k-1)\} + p_{\pi_i k}, \quad i = 2, \ldots, n, \quad k = 2, \ldots, \mathrm{m}, \tag{4}$$

Then, the makespan is given by

$$C_{\max}(\pi) = C(\pi_n, m). \tag{5}$$

Therefore, the problem turns into finding a permutation π^* in the set of all permutations Π such that

$$C_{\max}(\pi^*) \le C(\pi_n, m) \quad \forall \pi \in \Pi. \tag{6}$$

3. SOCIAL SPIDER OPTIMIZATION

3.1 BACKGROUND

The social spider optimization (SSO) suggested by Cuevas et al. [30] is a recent swarm intelligence algorithm inspired by the collaborative behavior of social spider colony. This behavior can be summarized in the following way. A social spider colony is composed of two essential components: spiders and communal web. The spiders are grouped into two different categories: males and females. Based on its gender, each spider collaborates in different tasks such as building and maintaining the communal web, prey capturing, and mating. Interactions among spiders are either direct or indirect. Direct interactions involve physical contact, such as mating. Indirect interactions take place by using the communal web as a medium of communication that transfers important information, such as the size of the trapped preys and characteristics of the neighboring members. This information, which is encoded as small vibrations, is a crucial portion for the mutual coordination among the spiders. The intensity of these vibrations is dependent on the locations and weights of the spiders that generate them [30].

This collaborative behavior can be utilized for solving optimization problems by simulation as follows. SSO imagines the communal web as the search space. The location of a spider in the communal web symbolizes a solution of the problem in the search space. Each spider is given a value for its weight that depends on the fitness of the solution that is represented by it. Unlike most of the existent swarm algorithms, SSO models two different search agents (spiders): males for performing extensive exploitation and females for performing efficient exploration. This allows not only to imitate the collaborative behavior of the colony in a better realistic way, but also to utilize computational operators that can delay the premature convergence and somehow strike an exploration–exploitation balance [30]. The search process is controlled by three operators: the female cooperative operator that changes the locations of females, the male cooperative operator that changes the locations of males, and the mating operator that produces new spiders that are located on new locations in the search space [30]. Figure 1 describes the general framework of the proposed SSO. As it can be noticed, a supplementary step has been added to the classical SSO, in which the locations of the spiders that exist in the continuous space are converted to their equivalent locations in the combinatorial space. What follows is the mathematical modelling of the proposed SSO [30].

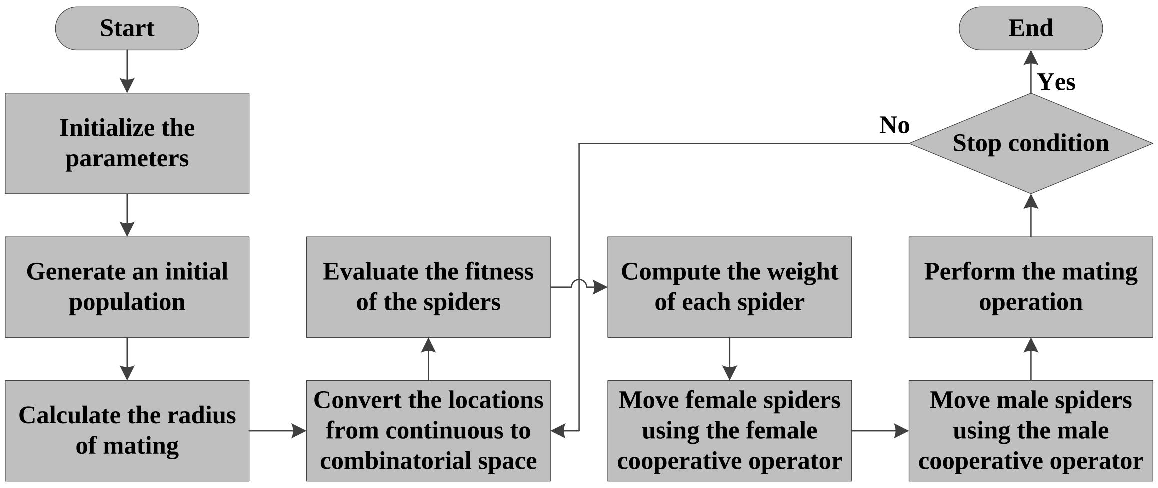
Start
Initialize the parameters
Generate an initial population
Calculate the radius of mating
Convert the locations from continuous to combinatorial space
Evaluate the fitness of the spiders
Compute the weight of each spider
Move female spiders using the female cooperative operator
Move male spiders using the male cooperative operator
Perform the mating operation
Stop condition
No
Yes
End

Figure 1. The proposed SSO.

3.2 GENDER ASSIGNATION

The first step in SSO is determining the number of female and male spiders. Since social spider colonies are highly female-biased ones, the number of females N_f is selected at random in the range 65–90% of the population. Let N be the total number of spiders, then N_f is calculated using the following formula.

$$N_f = floor[(0.9 - \text{rand} \cdot 0.25) \cdot N] \tag{7}$$

where rand is a random number in the range (0, 1), and the floor function is used to convert a real number into an integer number. The number of male spiders N_m is calculated as the complement between N and N_f using the following formula.

$$N_m = N - N_f \tag{8}$$

For that reason, the entire population of spiders S is split into female spiders group $F = \{f_1, f_2, \ldots f_{N_f}\}$ and male spiders group $M = \{m_1, m_2, \ldots m_{N_m}\}$, where $S = F \cup M (S = \{s_1, s_2, \ldots s_N\})$, such that $S = \{s_1 = f_1, s_2 = f_2, \ldots, s_{N_f} = f_{N_f}, s_{N_f+1} = m_1, s_{N_f+2} = m_2, \ldots, s_N = m_{N_m}\}$.

3.3 COLONY INITIALIZATION

The locations of all spiders are initialized at random. Each spider location, f_i or m_i, is an n-dimensional vector that represents the optimization variables, where n is the number of jobs in PFSP. The n components of each vector are uniformly distributed between the predefined lower initial parameter bound p_j^{low} and upper parameter bound p_j^{high}. These values are calculated using the following formulas.

$$f_{i,j}^0 = p_j^{low} + rand(0,1) \cdot \left(p_j^{high} - p_j^{low}\right) \quad i = 1,2,\ldots,N_f;\ j = 1,2,\ldots,n \tag{9}$$

$$m_{k,j}^0 = p_j^{low} + rand(0,1) \cdot \left(p_j^{high} - p_j^{low}\right) \quad k = 1,2,\ldots,N_m;\ j = 1,2,\ldots,n \tag{10}$$

where j refers to the index of a variable, i refers to the index of a female individual, k refers to the index of a male individual, the value 0 indicates that the individuals belong to the initial population, and rand(0,1) is a function used to generate a random number in the range (0,1). Hence, $f_{i,j}$ is the jth job of the ith female spider and $m_{k,j}$ is the jth job of the kth male spider.

3.4 CONTINUOUS TO COMBINATORIAL TRANSFORMATION

Since SSO works only on real numbers encoding, the random keys discretization method, which was initially proposed in [31], is utilized to transfer the locations of spiders from the continuous space to their corresponding locations in the combinatorial space.

3.5 WEIGHT CALCULATION AND ASSIGNATION

In the biological metaphor, the size of a spider is the characteristic that estimates its ability to do its assigned duties well. In SSO, each individual *i* of the population *S* is assigned a weight w_i that depends on the quality of the solution it symbolizes (irrespective of gender). The weight of each spider is calculated by using the following formula.

$$w_i = \frac{C_{max}(s_i) - C_{max}^{worst}}{C_{max}^{best} - C_{max}^{worst}} \tag{11}$$

where $C_{max}(s_i)$ is the makespan value obtained by decoding the permutation of jobs that corresponds to the location of the spider s_i, and C_{max}^{best} and C_{max}^{worst} are the makespan values of the best and worst individuals in the population, respectively.

3.6 MODELING OF THE VIBRATIONS

The vibrations observed by a spider depend on the distance and weight of the spiders that generate them. In SSO, the vibrations observed by spider *i* as a result of the information sent by another spider *j* are modeled according to the following formula.

$$Vib_{ij} = w_j \cdot e^{-d_{i,j}^2} \tag{12}$$

where $d_{i,j}$ is the Euclidean distance between the two spiders. SSO considers that each spider *i* is supposed to be able to detect vibrations from three other spiders. These spiders are the closest one that has a higher weight $Vibc_i$, the best spider in the colony $Vibb_i$, and the nearest female spider $Vibf_i$.

3.7 FEMALE COOPERATIVE OPERATOR

Female spiders are commonly attracted to the other (male or female) spiders in accordance with their vibrations transmitted over the communal web. Strong vibrations are generated by either big spiders or other neighboring spiders lying nearby the spider that is perceiving them. The decision of attraction or repulsion is made according to an internal state which is affected by several factors such as reproduction cycle, curiosity, and other random phenomena. This behavior is modeled by the female cooperative operator which is defined as follows.

$$f_i^{k+1} = \begin{cases} f_i^k + \alpha \cdot Vibc_i \cdot \left(s_c - f_i^k\right) + \beta . Vibb_i \cdot \left(s_b - f_i^k\right) \\ \qquad + \delta \cdot \left(\text{rand} - \frac{1}{2}\right) \quad if\ r_m < PF \\ f_i^k - \alpha \cdot Vibc_i \cdot \left(s_c - f_i^k\right) - \beta . Vibb_i \cdot \left(s_b - f_i^k\right) \\ \qquad + \delta \cdot \left(\text{rand} - \frac{1}{2}\right) \quad if\ r_m > PF \end{cases} \tag{13}$$

where α, β, δ, r_m, and rand are random numbers in the range (0, 1), k represents the iteration counter, PF represents the threshold value used for determining whether an attraction or repulsion movement is produced, s_c is the closest spider to spider *i* that has a higher weight, and s_b is the best spider in the population.

3.8 MALE COOPERATIVE OPERATOR

Male spiders consider themselves as a group of alpha males which dominate the colony resources. Hence, the males are divided into two sets: dominant and non-dominant males. Dominant males usually have superior fitness attributes to non-dominant males. In addition, dominant males are enticed to the nearest female spiders in the communal web. On the hand, non-dominant males tend to localize on the center of the set of males as a strategy to benefit from the resources left over from the dominant males. For implementing such phenomena, the set of males $M = \{m_1, m_2, \dots m_{N_m}\}$ is sorted according to their weight values in increasing order. The male spider that is located in the middle, whose weight value is w_{N_f+m}, is treated as the median male spider. The male spiders whose weight values are bigger the median value are treated as members of the set of dominant males *D*, and the rest of males are treated as members of the non-dominant set *ND*. In accordance to this, the variation of locations for the male spiders is modeled by the following formula.

$$m_i^{k+1} = \begin{cases} m_i^k + \alpha \cdot Vibf_i \cdot (s_f - m_i^k) + \delta \cdot \left(\text{rand} - \frac{1}{2}\right) \\ \qquad if\ w_{N_f+i} > w_{N_f+m} \\ m_i^k + \alpha \cdot \left(\frac{\sum_{h=1}^{N_m} m_h^k \cdot w_{N_f+h}}{\sum_{h=1}^{N_m} w_{N_f+h}} - m_i^k\right) \\ \qquad if\ w_{N_f+i} \le w_{N_f+m} \end{cases} \tag{14}$$

where s_f is the nearest female spider to the male *i*, and $\sum_{h=1}^{N_m} m_h^k \cdot w_{N_f+h} / \sum_{h=1}^{N_m} w_{N_f+h}$ is the weighted mean of the set of male spiders *M*.

3.9 MATING OPERATOR

Mating is performed by dominant males and females. A dominant male spider m_g ($m_g \in D$) can mate with a set of female spiders E^g if they exist within a specific radius *r* (range of mating). This radius, which depends on the size of the search space, is calculated by the following formula.

$$r = \frac{\sum_{j=1}^{n}(p_j^{high} - p_j^{low})}{2 \cdot n} \tag{15}$$

If $E^g = \emptyset$, the mating process is revoked. Otherwise, mating occurs and a new brood s_{new} is generated by taking in consideration all elements of the set T^g which is the union $m_g \cup E^g$. During the mating process, the weight of each element of the set T^g controls the probability of its impact on the new brood. The members with higher weight values have higher probabilities to impact the new spider than those with lighter weight values. The impact probability Ps_i of each member is calculated by the roulette wheel method, which is defined as follows.

$$Ps_i = \frac{w_i}{\sum_{j \in T^g} w_j} \tag{16}$$

where $i \in T^g$. When a new spider is born, it is immediately compared to the spider holding the worst weight of the whole colony. If the new spider is superior to the worst spider, it replaces it. Otherwise, the worst spider is kept and the new spider is neglected.

4. COMPUTATIONAL RESULTS

SSO has been implemented in C++. The tests have been run on a personal computer with 3.4 GHz CPU and 8 GB RAM. Two well-known benchmark sets were used for the evaluation. The first set is Reeves's set [32]. This set is composed of 21 instances that are divided into 7 equal subsets of different sizes. These sizes range from 20 jobs and 5 machines up to 75 jobs and 20 machines. The second set is Taillard's set [33]. This set is composed of 120 instances that are divided into 12 equal subsets of different sizes. These sizes range from 20 jobs and 5 machines up to 500 jobs and 20 machines. In order to tune the parameters, preliminary experiments have been done on 19 instances selected randomly from the two benchmark sets (one of each subset). Consequently, the parameters have been set as follows: $N=100$, $PF=0.7$, α, β, δ, and r_m are as assigned random values between zero and one, and the maximum number of iterations is 10000.

SSO was compared with conventional and best-so-far metaheuristics. The metaheuristics that were compared using Reeves's set are chaotic local search based bacterial foraging algorithm (CLS-BFO) [11], shuffled complex evolution algorithm with opposition-based learning (SCE-OBL) [7], and Genetic algorithm integrated with artificial chromosomes (ACGA) [34]. The metaheuristics that were compared using Taillard's set are memetic algorithm with novel semi-constructive evolution operators (MASC) [8] which is one of the best-so-far approaches for the problem, hybrid whale optimization algorithm based on local search strategy (HWA) [35], and self-guided differential evolution with neighborhood search (NS-SGDE) [16]. SSO was run 10 independent times and the best solution (BS) among them was considered for the comparison. Table 1 presents the results on Reeves's set. It lists instance name, instance size (number of jobs * number of machines), best known solution (BKS), BS of each algorithm. From Table 1, it can be seen that SSO is able to obtain better solutions than the others on most of the instances. However, to make a precise comparison, the relative error of BS for each instance (*PE*), and the average of *PE* for the whole set of instances (*APE*) were calculated for each algorithm as follows.

$$PE = 100 \times \left(\frac{BS-BKS}{BKS}\right) \qquad (17)$$

$$APE = \left(\sum_{i=1}^{21}\left(\frac{BS_i-BKS_i}{BKS_i}\right) \times 100\right)/21 \qquad (18)$$

Table 2 presents the results. It lists *APE* values for SSO and the other algorithms (OA), and the percentage improvement (*PI*) achieved by SSO in *APE* values with respect to each of the other algorithms, which was calculated as follows.

$$PI = 100 \times (OA_{APE} - SSO_{APE})/OA_{APE} \qquad (19)$$

From Table 2, it can be seen that SSO has the lowest *APE* value and produces relative improvements to all the other algorithms. This shows that SSO is an effective approach since the compared works are hybrid methods. To further verify the effectiveness of SSO, the one-way ANOVA test was applied on the *PE* values. Figure 2 shows the results. From Figure 2, it can be seen that SSO outperforms the compared algorithms, and the resulting *p*-value is 0,009 which implies that the algorithms are significantly statistically different with each other.

Table 1. The computational results on Reeves's set.

Instance	*n*m*	BKS	SSO	SCE-OBL	CLS-BFO	ACGA
Rec1	20x5	1245	1247	1249	1249	1249
Rec3	20x5	1109	1109	1111	1111	1109
Rec5	20x5	1242	1245	1245	1245	1245
Rec7	20x10	1566	1566	1584	1584	1566
Rec9	20x10	1537	1537	1545	1545	1537
Rec11	20x10	1431	1431	1431	1449	1431
Rec13	20x15	1930	1935	1963	1968	1935
Rec15	20x15	1950	1968	1993	1993	1950
Rec17	20x15	1902	1923	1944	1954	1911
Rec19	30x10	2093	2117	2156	2139	2099
Rec21	30x10	2017	2017	2064	2059	2046
Rec23	30x10	2011	2030	2067	2073	2021
Rec25	30x15	2513	2566	2584	2638	2545
Rec27	30x15	2373	2397	2445	2443	2396
Rec29	30x15	2287	2333	2364	2408	2304
Rec31	50x10	3045	3104	3179	3180	3105
Rec33	50x10	3114	3118	3154	3187	3140
Rec35	50x10	3277	3277	3281	3292	3277
Rec37	75x20	4890	5096	5327	5422	5193
Rec39	75x20	5043	5185	5391	5465	5276
Rec41	75x20	4910	5135	5334	5436	5208

Table 2. *APE* of SSO and the other works on Reeves's set.

Algorithm	***APE***		***PI***
	OA (%)	**SSO (%)**	**SSO (%)**
CLS-BFO	2,675	1,123	58
SCE-OBL	3,255	1,123	65
ACGA	1,247	1,123	10

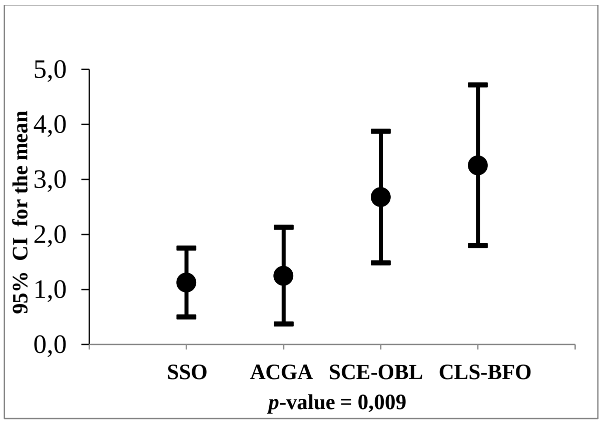

Figure 2. Means and 95% confidence intervals on Reeves's set.

Table 3. The computational results on ST1.

Instance	*n*m*	BKS	SSO	NS-SGDE	HWA	MASC
Ta001	20x5	1278	1282	1278	1278	1278
Ta002	20x5	1359	1359	1359	1359	1359
Ta003	20x5	1081	1088	1081	1081	1081
Ta004	20x5	1293	1300	1293	1293	1293
Ta005	20x5	1235	1237	1235	1235	1235
Ta006	20x5	1195	1195	1195	1195	1195
Ta007	20x5	1239	1243	1239	1239	1239
Ta008	20x5	1206	1206	1206	1206	1206
Ta009	20x5	1230	1231	1230	1230	1230
Ta010	20x5	1108	1108	1108	1108	1108
Ta011	20x10	1582	1598	1582	1582	1582
Ta012	20x10	1659	1682	1659	1659	1659
Ta013	20x10	1496	1513	1496	1496	1496
Ta014	20x10	1377	1395	1377	1377	1377
Ta015	20x10	1419	1440	1419	1419	1419
Ta016	20x10	1397	1404	1397	1397	1397
Ta017	20x10	1484	1493	1484	1484	1484
Ta018	20x10	1538	1555	1538	1538	1538
Ta019	20x10	1593	1606	1593	1593	1593
Ta020	20x10	1591	1611	1591	1591	1591
Ta021	20x20	2297	2329	2297	2297	2297
Ta022	20x20	2099	2125	2099	2099	2099
Ta023	20x20	2326	2350	2326	2326	2326
Ta024	20x20	2223	2244	2223	2223	2223
Ta025	20x20	2291	2309	2291	2291	2291
Ta026	20x20	2226	2244	2228	2226	2226
Ta027	20x20	2273	2296	2273	2273	2273
Ta028	20x20	2200	2229	2200	2200	2200
Ta029	20x20	2237	2252	2237	2237	2237
Ta030	20x20	2178	2195	2178	2178	2178
Ta031	50x5	2724	2724	2724	2724	2724
Ta032	50x5	2834	2839	2834	2834	2834
Ta033	50x5	2621	2621	2621	2621	2621
Ta034	50x5	2751	2753	2751	2751	2751
Ta035	50x5	2863	2863	2863	2863	2863
Ta036	50x5	2829	2832	2829	2829	2829
Ta037	50x5	2725	2725	2725	2725	2725
Ta038	50x5	2683	2703	2683	2683	2683
Ta039	50x5	2552	2561	2552	2552	2552
Ta040	50x5	2782	2782	2782	2782	2782
Ta041	50x10	2991	3053	3021	3021	3024
Ta042	50x10	2867	2938	2896	2891	2882
Ta043	50x10	2839	2890	2888	2869	2852
Ta044	50x10	3063	3071	3075	3063	3063
Ta045	50x10	2976	3024	3027	3001	2982
Ta046	50x10	3006	3050	3029	3006	3006
Ta047	50x10	3093	3133	3124	3126	3099
Ta048	50x10	3037	3046	3055	3046	3038
Ta049	50x10	2897	2927	2928	2897	2902
Ta050	50x10	3065	3131	3092	3078	3077
Ta051	50x20	3850	3974	3916	3876	3889
Ta052	50x20	3704	3808	3744	3715	3720
Ta053	50x20	3640	3772	3702	3653	3667
Ta054	50x20	3720	3849	3793	3755	3754
Ta055	50x20	3610	3746	3677	3649	3644
Ta056	50x20	3681	3795	3743	3703	3708
Ta057	50x20	3704	3835	3784	3723	3754
Ta058	50x20	3691	3829	3757	3704	3711
Ta059	50x20	3743	3870	3795	3763	3772
Ta060	50x20	3756	3875	-	3767	3778

Table 4. The computational results on ST2.

Instance	*n*m*	BKS	SSO	NS-SGDE	HWA	MASC
Ta061	100x5	5493	5493	5493	5493	5493
Ta062	100x5	5268	5284	5283	5268	5268
Ta063	100x5	5171	5193	5182	5175	5175
Ta064	100x5	5014	5023	5018	5018	5014
Ta065	100x5	5250	5250	5253	5250	5250
Ta066	100x5	5135	5137	5135	5135	5135
Ta067	100x5	5246	5259	5246	5246	5246
Ta068	100x5	5094	5106	5094	5094	5094
Ta069	100x5	5448	5467	5448	5448	5448
Ta070	100x5	5322	5338	5322	5324	5322
Ta071	100x10	5770	5787	5784	5776	5770
Ta072	100x10	5349	5379	5362	5362	5349
Ta073	100x10	5676	5691	5691	5691	5677
Ta074	100x10	5781	5849	5826	5825	5781
Ta075	100x10	5467	5513	5503	5491	5467
Ta076	100x10	5303	5328	5308	5308	5304
Ta077	100x10	5595	5662	5610	5608	5596
Ta078	100x10	5617	5695	5630	5630	5625
Ta079	100x10	5871	5916	5882	5891	5875
Ta080	100x10	5845	5903	5881	5848	5845
Ta081	100x20	6202	6377	6360	6280	6257
Ta082	100x20	6183	6360	6278	6278	6223
Ta083	100x20	6271	6450	6405	6368	6325
Ta084	100x20	6269	6393	6394	6350	6303
Ta085	100x20	6314	6477	6452	6377	6380
Ta086	100x20	6364	6544	6461	6430	6431
Ta087	100x20	6268	6439	6385	6354	6306
Ta088	100x20	6401	6603	6496	6515	6472
Ta089	100x20	6275	6451	6428	6396	6330
Ta090	100x20	6434	6625	6538	6527	6456
Ta091	200x10	10862	10947	10887	10885	10872
Ta092	200x10	10480	10542	10555	10512	10487
Ta093	200x10	10922	11005	10980	10965	10922
Ta094	200x10	10889	10939	10917	10889	10889
Ta095	200x10	10524	10537	10537	10524	10526
Ta096	200x10	10326	10377	10357	10375	10330
Ta097	200x10	10854	10908	10929	10868	10868
Ta098	200x10	10730	10798	10798	10751	10731
Ta099	200x10	10438	10478	10465	10465	10454
Ta100	200x10	10675	10758	10727	10727	10680
Ta101	200x20	11195	11418	11468	11335	11280
Ta102	200x20	11203	11488	11487	11517	11272
Ta103	200x20	11281	11559	11549	11481	11378
Ta104	200x20	11275	11465	11553	11405	11376
Ta105	200x20	11259	11444	11438	11374	11310
Ta106	200x20	11176	11421	11445	11335	11265
Ta107	200x20	11360	11593	11596	11438	11430
Ta108	200x20	11334	11597	11592	11530	11398
Ta109	200x20	11192	11457	11485	11439	11266
Ta110	200x20	11288	11567	11607	11499	11355
Ta111	500x20	26059	26493	26420	26388	26187
Ta112	500x20	26520	26953	26942	26714	26779
Ta113	500x20	26371	26787	26729	26648	26496
Ta114	500x20	26456	26817	26751	26656	26618
Ta115	500x20	26334	26698	26643	26579	26500
Ta116	500x20	26477	26874	26832	26666	26647
Ta117	500x20	26389	26691	26609	26594	26529
Ta118	500x20	26560	26913	26925	26711	26772
Ta119	500x20	26005	26425	26326	26228	26223
Ta120	500x20	26457	26905	26766	26695	26617

For the clarity of presentation, the benchmark set of Taillard was divided into two subsets. The first subset is named ST1 and contains the instances where the number of operations <=100. The second subset is named ST2 and contains the instances where the number of operations >100. Table 3 and Table 4 show BS obtained by the compared works on these two subsets.

From Table 3 and Table 4, it can be seen that the performance of SSO deteriorates on this set. However, in order to provide a more thorough comparison, the one-way ANOVA test was applied on the *PE* values over the whole set of instances. Figure 3 shows the results. From Figure 3, it can be noticed that MASC performs better than all the compared works. This is because MASC combines the complementary strengths of population-based (global search), constructive methods, and single-point (local search) methods. The four works can be sorted according to the resulting *APE* values from the best to the worst as follows: MAC 0,278, HWA 0,460, NS-SGDE 0,750, SSO 1,268. It can be also noticed that the resulting *p*-value is 3,83E-22 which indicates that there is a statistically significant difference between the compared algorithms.

The deterioration of SSO performance on this set can be due to two reasons. First, Taillard's set is more difficult to solve than Reeves's set. Second, SSO didn't use any problem-specific operators or external single-point mechanisms. Therefore, it cannot intensify the search in the promising areas of the search space, i.e. the neighborhood of good solutions.

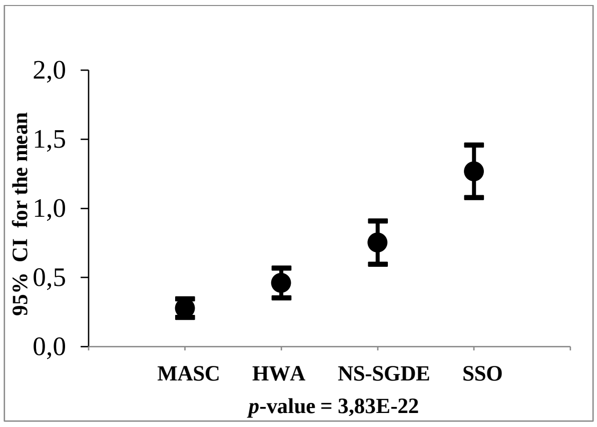

Figure 3. Means and 95% confidence intervals on Taillard's set.

5. CONCLUSION AND FUTURE WORKS

In this work, SSO algorithm is proposed to solve the permutation flow shop problem with makespan minimization. The aim is to investigate the effectiveness of SSO in solving this combinatorial problem. The well-known benchmark sets of Reeves and Taillard were used for the evaluation. Six other conventional and best-so-far algorithms were used for the comparison. The computational results show that SSO outperforms three of them although they are hybrid methods. However, the results also show that it fails to compete with the best-so-far algorithms. This is due to the fact that the best-so-far approaches are memetic algorithms that combine the advantages of population-based and single-point algorithms. Therefore, it is expected that SSO will perform very well if it is hybridized with a single-point algorithm, and this hybridization could be one of the future works of this research.

REFERENCES

[1] V. Fernandez-Viagas, R. Ruiz, J. M. Framinan, “A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation”, *Eur. J. Oper. Res.*, vol. 257, no. 3, pp. 707-721, Mar. 2017.

[2] Q. Lin, L. Gao, X. Li, C. Zhang, ”A hybrid backtracking search algorithm for permutation flow-shop scheduling problem”, *Comput. Ind. Eng.*, vol. 85, pp. 437-446, July. 2015.

[3] N. A. Alawad, B. H. Abed-alguni, ”Discrete Jaya with refraction learning and three mutation methods for the permutation flow shop scheduling problem”, *J Supercomput*, pp. 1-22, July. 2021.

[4] P. Wu, Q. Yang, W. Chen, B. Mao, H. Yu, ”An Improved Genetic-Shuffled Frog-Leaping Algorithm for Permutation Flowshop Scheduling”, *Complexity*, 2020.

[5] L. Libralesso, P. A. Focke, A Secardin, V. Jost, “Iterative beam search algorithms for the permutation flowshop”, *arXiv preprint arXiv*, Sep. 2020.

[6] C. Y. Hsu, P. C. Chang, M. H. Chen, “A linkage mining in block-based evolutionary algorithm for permutation flowshop scheduling problem”, Comput. Ind. Eng., vol. 83, pp. 159-171, May. 2015.

[7] F. Zhao, J. Zhang, J. Wang, C. Zhang, “A shuffled complex evolution algorithm with opposition-based learning for a permutation flow shop scheduling problem”, *Int. J. Computer Integr. Manuf.*, vol. 28, no. 11, pp. 1220-1235, Oct. 2015.

[8] M. Kurdi, ”A memetic algorithm with novel semi-constructive evolution operators for permutation flowshop scheduling problem”, *Appl. Soft Comput.*, vol. 94, Sept. 2020.

[9] M. Bessedik, F. B. S. Tayeb, H. Cheurfi, A. Bliz, ”An immunity-based hybrid genetic algorithms for permutation flowshop scheduling problems”, *Int. J. Adv. Manuf. Syst.*, vol. 85. no 9-12, pp. 2459-2469, Aug. 2016.

[10] F. B. S. Tayeb, M. Bessedik, M. Benbouzid, M. Benbouzid, H. Cheurfi, A. Blizak, “Research on Permutation Flow-shop Scheduling Problem based on Improved Genetic Immune Algorithm with vaccinated offspring”, *Procedia Comput. Sci.*, vol. 112, pp. 427-436, 2017.

[11] F. Zhao, Y. Liu, Z. Shao, X. Jiang, C. Zhang, J. Wang, “A chaotic local search based bacterial foraging algorithm and its application to a permutation flow-shop scheduling problem”, *Int. J. Computer Integr. Manuf.*, vol. 29, no. 9, pp. 962-981, 2016.

[12] Ö. Tosun, M. K. Marichelvam, “Hybrid bat algorithm for flow shop scheduling problems”, *Int. J. Math. Oper.*, vol. 9, no. 1, pp. 125-138, 2016.

[13] S. Deb, Z. Tian, S. Fong, R. Tang, R. Wong, N. Dey, “Solving permutation flow-shop scheduling problem by rhinoceros search algorithm”, *Soft Comput.*, vol. 22, no. 18, pp. 6025-6034, 2018.

[14] J. Lin, “A hybrid discrete biogeography-based optimization for the permutation flow shop scheduling problem” *Int. J. Prod. Res.*, vol. 54, no. 16, pp. 4805-4814, 2016.

[15] V. Santucci, M. Baioletti, A. Milani, ”Solving permutation flowshop scheduling problems with a discrete differential evolution algorithm”, *AI Commun.*, vol. 29, no. 2, pp. 269-286, 2016.

[16] W. Shao, D. Pi, “A self-guided differential evolution with neighborhood search for permutation flow shop scheduling”, *Expert Syst. Appl.*, vol. 51, pp. 161-176, June. 2016.

[17] F. Zhao, Y. Liu, Y. Zhang, W. Ma, C. Zhang, “A hybrid harmony search algorithm with efficient job sequence scheme and variable neighborhood search for the permutation flow shop scheduling problems”, *Eng. Appl. Artif. Intell.*, vol. 65, pp. 178-199, Oct. 2017.

[18] H. Wang, W. Wang, H. Sun, Z. Cui, S. Rahnamayan, S. Zeng, “A new cuckoo search algorithm with hybrid strategies for flow shop scheduling problems”, *Soft Comput.*, vol. 21, no. 15, pp. 4297-4307, 2017.

[19] K .Govindan, R. Balasundaram, N. Baskar, P. Asokan, “A hybrid approach for minimizing makespan in permutation flowshop scheduling”, *J. Syst. Sci. Syst. Eng.*, vol. 26, no. 1, pp. 50-76, 2017.

[20] J. Dubois-Lacoste, F. Pagnozzi, T. Stützle, “An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem”, *Comput. Oper. Res.*, vol. 81, pp.160-166, May. 2017.

[21] M. K. Marichelvam, Ö. Tosun, M. Geetha, “Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time”, *Appl. Soft Comput*., vol. 55, no. 82-92, 2017.

[22] Y. Zhang, Y. Yu, S. Zhang, Y. Luo, L. Zhang, “Ant colony optimization for Cuckoo Search algorithm for permutation flow shop scheduling problem”, *Syst. Sci. Control. Eng.*, vol. 7, no. 1, pp. 20-27, 2019.

[23] S. M. Lim, K. Y. Leong. “A brief survey on intelligent swarm-based algorithms for solving optimization problems”, in *Nature-inspired Methods for Stochastic, Robust and Dynamic Optimization*, 2018, ch. 4, pp 47–61.

[24] S. Kavitha, P. Venkumar, N . Rajini, P. Pitchipoo, “An Efficient Social Spider Optimization for Flexible Job Shop Scheduling Problem”, *J. Adv. Manuf. Syst.*, vol. 17, no. 2, pp. 181-196, 2018.

[25] Y. Wang, L. Zhu, J. Wang, J. Qiu, “An improved social spider algorithm for the Flexible Job-Shop Scheduling Problem”, in *ICEDIF*, Harbin, China, 2015, pp. 157-162.

[26] M. Kurdi, “A Social Spider Optimization Algorithm for Hybrid Flow Shop Scheduling with Multiprocessor Task”, in *12th NCMCONFERENCES*, Ankara, Turkey, 2018. Available at SSRN 3301792.

[27] G. Zhang, K. Xing, “Memetic social spider optimization algorithm for scheduling two-stage assembly flowshop in a distributed environment”, *Comput. Ind. Eng.*, vol. 125, pp. 423-433, Nov. 2018.

[28] G. Zhou, Y. Zhou, R. Zhao, “Hybrid social spider optimization algorithm with differential mutation operator for the job-shop scheduling problem”, *J. Ind. Manag*., vol. 17, no. 2, pp. 533-548, Mar. 2021.

[29] G. I. Zobolas, C. D. Tarantilis and G. Ioannou, “Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm”, *Comput. Oper. Res.*, vol. 36, no. 4, pp. 1249-1267, Apr. 2009.

[30] E. Cuevas, M. Cienfuegos, D. Zaldívar, M. Pérez-Cisneros, ”A swarm optimization algorithm inspired in the behavior of the social-spider”, *Expert Syst. Appl.*, vol 40, no. 16, pp. 6374-6384, Nov. 2013.

[31] J. C. Bean, “Genetic algorithms and random keys for sequencing and optimization”, *ORSA J. Comput.*, vol. 6. no. 2, pp.154-160, 1994.

[32] C. R. Reeves, ”A genetic algorithm for flowshop sequencing”, *Comput. Oper. Res*., vol. 22, no. 1, pp. 5-13, Jan. 1995.

[33] E. Taillard, "Benchmarks for basic scheduling problems”, *Eur. J. Oper. Res*., vol. 64, no. 2, pp. 278-285, 1993.

[34] P. C. Chang, S. H. Chen, C. Y. Fan, C. L. Chan, “Genetic algorithm integrated with artificial chromosomes for multi-objective flowshop scheduling problems”, *Appl. Math. Comput*., vol. 205, no. 2, pp. 550-561, Nov. 2008.

[35] M. Abdel-Basset, G. Manogaran, D. El-Shahat, S. Mirjalili, “A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem”, *Future Gener. Comput. Syst*. vol. 85, pp. 129-145, Aug. 2018.

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:98-107 (2021)
Received: October 7, 2021 Accepted: November 3, 2021
JSCAI
ISSN: 2717-8226

Implementation of Advanced PID Control Algorithm for SDOF System

[1] **Abdullah Turan** , [2] **Huseyin Aggumus**

[1,2]Department of Mechanical and Metal Technologies, Sirnak University, Sirnak, 73000 TURKEY

Corresponding author: First A. Author (e-mail: abdullahturan@sirnak.edu.tr).

ABSTRACT The maximum performance to be obtained by applying the Proportion-Integral-Derivative (PID) controller on a system depends on the optimum adjustment of its parameters. This study aims to present a design method for tuning the PID control parameters. In this method, PID controller design is made based on the optimal proportional gain from the system to the desired settling time and overshoot. The infrastructure of the technique is based on obtaining the other PID controller parameters by adjusting the optimum proportional gain (k_p), which minimizes the settling time in a stable loop and the error rate of the overshoot. Routh Hurwitz criterion is used to guarantee stability in the control system. The effectiveness of the proposed method is tested as an active control application of the PID controller on a single degree of freedom (SDOF) structural system. The efficiency of the PID controller designed with this method, which does not require the destruction of parameters and does not contains complex mathematical formulations, is proven by its successful suppression of SDOF structural system responses.

KEYWORDS: Active control; PID controller; SDOF system; structural control.

1. INTRODUCTION

Control of structural systems is a current and vital issue. Damage to any part of a structure under the influence of external disturbances, such as crack formation in its beams [1,2], may damage the entire system. For this reason, control applications are one of the most effective methods to protect the structures from damage and the safety of the occupants. Although there are many control applications in the process control industry, PID control is widely used and maintains its importance due to its low cost, simple structure, and wide application area [3]. Therefore, improvements in the design methods of the PID controller will increase the performance of this controller. In general, the PID controller, which is obtained with an efficient design method, should respond optimally to the design features and be robust against uncertainties. For the PID controller's performance on the applied system to be at the maximum level, its parameters must be set correctly. The Ziegler and Nichols method is the oldest method used to set PID control parameters [4]. However, in the performance of the PID control obtained with this method, negative effects such as high settling time and overshoot arise. Therefore, the obtained answers need to be improved. In another study [5], in which a different method was proposed for the proportional gain value, it was determined that negative responses were obtained for time delay systems. Other standard methods in the literature include; gain and phase margin method [6,7], Astrom and Haggland method [8], optimization method [9,10], direct synthesis method [11], the weighted geometric center method [12,13]. In addition, many studies suggest new control methods [14-18]. For example, PID controller design using stability limit position matching [19], near-optimal PID controller design with interval arithmetic approach [20], PID controller design with numerical optimization approach [21] are suggested methods.

In this study, a new approach is discussed to determine the optimum PID parameters. In the proposed method, first of all, the desired settling time and overshoot values are defined in the closed-loop system response, then the proportional gain k_p, which will minimize the error rates of the expected settling time and overshoot values from the system, is determined with a loop formed in the stable area. Based on the obtained k_p value, the other parameters of the PID controller, k_i and k_d, are calculated. This proposed simple adjustment method has some advantages over other methods in the literature. The solution processes are simple as there is no need for complex mathematical equations, and there is a high chance of getting maximum efficiency as optimization occurs. The efficiency of the PID controller obtained with the proposed method in this study is tested on an SDOF structural system.

2. EQUATION OF MOTION

Control of structural systems is an area that remains popular today. As a result, both single-degree-of-freedom (SDOF) and multi-degree-of-freedom (MDOF) system models have been frequently used in studies. In this study, the SDOF structural system used to test the designed control performance is shown in Figure 1.

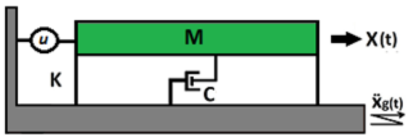

Figure 1. SDOF system

The control force acts on the system from outside. The equation of motion of the system is seen in Eq. (1).

$$M\ddot{x}(t) + C\dot{x}(t) + Kx(t) = -f_u(t) - M\ddot{x}_g \quad (1)$$

Here, M, C, and K denote the system's mass, damping, and stiffness values, respectively. f_u is the control force and $\ddot{x}_g$ is the acceleration excitation of the system. Parameter of the system are M= 107.5 kg, K= 145152 N/m, C= 6.7646 N.s/m.

3. DETERMINATION of PID PARAMETERS and CONTROLLER DESIGN

Both semi-active [22, 23] and active control [24] applications are widely studied in structural systems. Active control is an indispensable choice in obtaining the best performance from these systems. In this study, the active control of the SDOF structural system is investigated by obtaining the optimum parameters of the PID controller with a new approach. The PID control method remains essential because it has been widely used and can be easily applied to systems [25]. In the most general case, the block diagram of the unit feedback control system is shown in Figure 2.

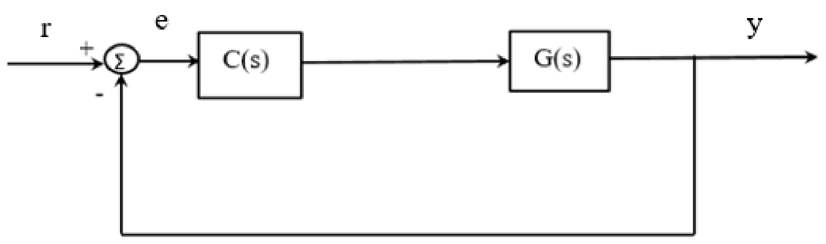

Figure 2. Feedback control system

Here, C(s) given in Eq. (2) and G(s) shown in Eq. (3) represent the transfer function of the PID controller and SDOF system, respectively. Here r, y and e denote the input, output of the system and error rate, respectively.

$$C(s) = k_p + \frac{k_i}{s} + k_d s \tag{2}$$

$$G_s = \frac{G_N(s)}{G_D(s)} \tag{3}$$

By arranging Eq. (2), the controller equation, in general, is obtained as follows.

$$C(s) = \frac{(k_d s^2 + k_p s + k_i)}{s} \tag{4}$$

The closed-loop system T(s) is obtained as in Eq. (5) using Eq. (4).

$$T(s) = \frac{C(s)G(s)}{1+(C(s)G(s))} \tag{5}$$

T(s) is obtained by substituting Eq. (3) and Eq. (4) in Eq. (5) as follows.

$$T(s) = \frac{G_N(s)(k_d s^2 + k_p s + k_i)}{G_D(s)s + G_N(s)(k_d s^2 + k_p s + k_i)} = \frac{T_N(s)}{T_D(s)} \tag{6}$$

$T_D(s)$ is the characteristic equation of the system, and its degree is determined. Then, settling time t_s and overshoot M_p value are determined according to system performance. The damping rate of the system is calculated by Eq. (7), and its natural frequency is calculated by Eq. (8).

$$\zeta = \frac{-\log(M_p)}{\sqrt{\pi^2 + \log(M_p)^2}} \tag{7}$$

$$w_n = \frac{4}{\zeta t_s} \tag{8}$$

The target polynomial equation of the closed-loop system is as follows.

$$\Delta(s) = s^2 + 2\zeta w_n s + {w_n}^2 \tag{9}$$

Here, a residue polynomial (R(s)) must be defined since Δ(s) is of order 2. Also, the variable number of R(s) should be equal to the degree difference (m) between $T_D(s)$ and Δ(s).

$$R(s) = \begin{cases} s + a, & m = 1 \\ s^2 + a_1 s + a_2, & m = 2 \\ s^3 + a_1 s^2 + a_2 s + a_3, & m = 3 \\ s^n + a_1 s^{n-1} + a_2 s^{n-2} + \cdots a_n s^{n-m}, & m = n \end{cases} \tag{10}$$

The fact that the variables expressed in Eq. (10) are $a_1, a_2, a_3 \ldots a_n \in R$ allows it to cover all solutions and keep the system free of complexity. The condition to be satisfied here is that the coefficients of the product of Δ(s) and R(s) are equal to the system's characteristic equation coefficients, as seen in Eq. (11).

$$(\Delta(s)R(s))_{coeff} \equiv T_D(s)_{coeff} \tag{11}$$

In Eq. (11), it is seen that the number of variables is one more than the number of significant equations. Therefore, to equalize the number of equations, the number of variables is reduced by one by subtracting k_p from the variables (k_p, k_i, k_d, a_1, $a_2 \ldots a_N$). Then, by solving Eq. (11) again, it is ensured that the variables contain terms with k_p. As a result, the existence of a multiple-choice solution emerges. Initially, Δ(s) is chosen as stable. Also, the R(s) polynomial must also check for stability. For stability, it is sufficient that the variables (a_1, $a_2 \ldots a_N$) in R(s) are positive. As a result, the flowchart in Figure 2 is followed to set the PID parameters optimally.

After the variables are determined as positive, values are assigned to k_p in a specific interval and incremental cycle, and the variables (k_p, k_i, k_d, a_1, $a_2 \ldots a_N$) and the t_s and M_p values of the system are determined. The aim here is to obtain a value as close as possible to the expected (desired) value of t_s and M_p. Therefore, the error rates of t_s and M_p, which need to be normalized, are assigned the variables e_1 and e_2, respectively.

$$e_1 = \frac{M_p - M_{p_{ans}}}{M_p} \tag{12}$$

$$e_2 = \frac{t_s - t_{s_{ans}}}{t_s} \tag{13}$$

By combining Eq. (12) and Eq. (13), Eq. (14) is obtained to show a single error.

$$err = xe_1 + ye_2 \tag{14}$$

Here, x, y are the coefficients affecting the total error, and x and y values are selected according to the importance expected from the system and x+y=1. The err value obtained is also added to the loop and determined according to k_p. Finally, PID controller parameters k_p, k_i and k_d are accepted according to the err_{min} value specified due to the loop.

3.1 PID CONTROLLER DESIGN

To determine the effectiveness of the method proposed in Section 3, a comparison is made with the PID controller tuning using the Matlab-Simulink Toolbox. In simulations, settling time, overshoot, peak time, and rise time are considered evaluation criteria. The transfer function of the model of the system in Figure 1 is as follows.

$$G(s) = \frac{0.0093023}{s^2+0.062927s+1350.2512} \tag{15}$$

First of all, 0.01 % overshoot and 0.2 s settling time are expected performance values from the system. The residual polynomial is treated as in Eq. (17) (m=1).

$$R(s) = s + a \tag{16}$$

Therefore, the expression $a = 0.0004651*k_p + 60.19$ is obtained with the proposed method, and it is sufficient for the variable a to be positive for the stability criterion. Therefore, for k_p, the interval value $k_p = [-100000,10000000]$ is selected in increments of 0.1. A loop is created by selecting the total error value as in Equation 14.

The parameters of the PID controller, from which the err_{min} value is obtained in the performed cycle, are given in Table 1. By applying the PID-1 control created in Matlab-Simulink Toolbox and the PID-2 control obtained by the proposed method to the system, unit step responses in the closed-loop are given in Figure 3. In addition, the bode and root locus graphs of the system with the PID-2 controller are shown in Figure 4 and Figure 5, respectively. In the characteristic equation for the whole closed-loop, the roots are in the left half of the 's' plane. Therefore, the system is stable. In the unit step response with the designed PID-2 controller, a better (%) overshoot value and peak time are obtained as well as much better settling time. Performance criteria of PID-1 and PID-2 controllers are given in Table 1. It can be clearly said that the proposed PID-2 controller performs better than PID-1.

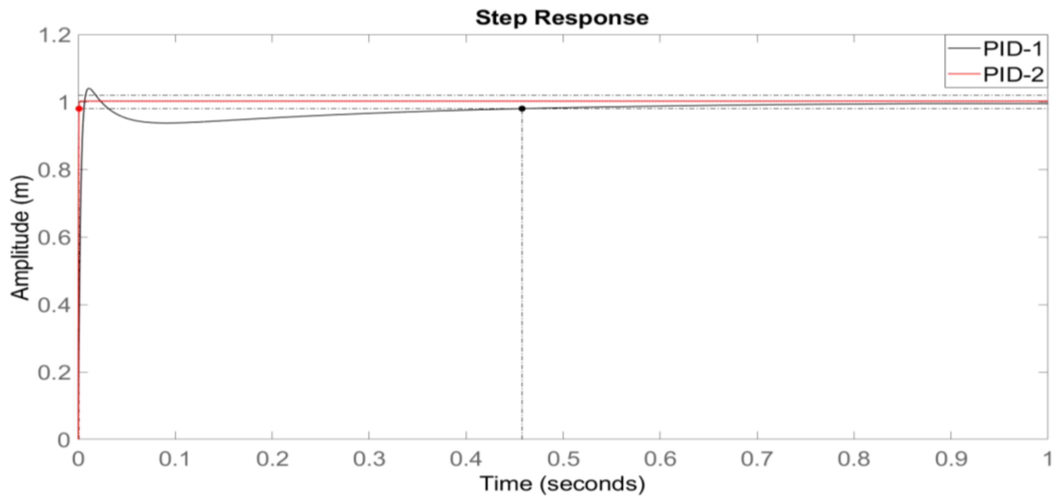

Figure 3. Step responses for the system.

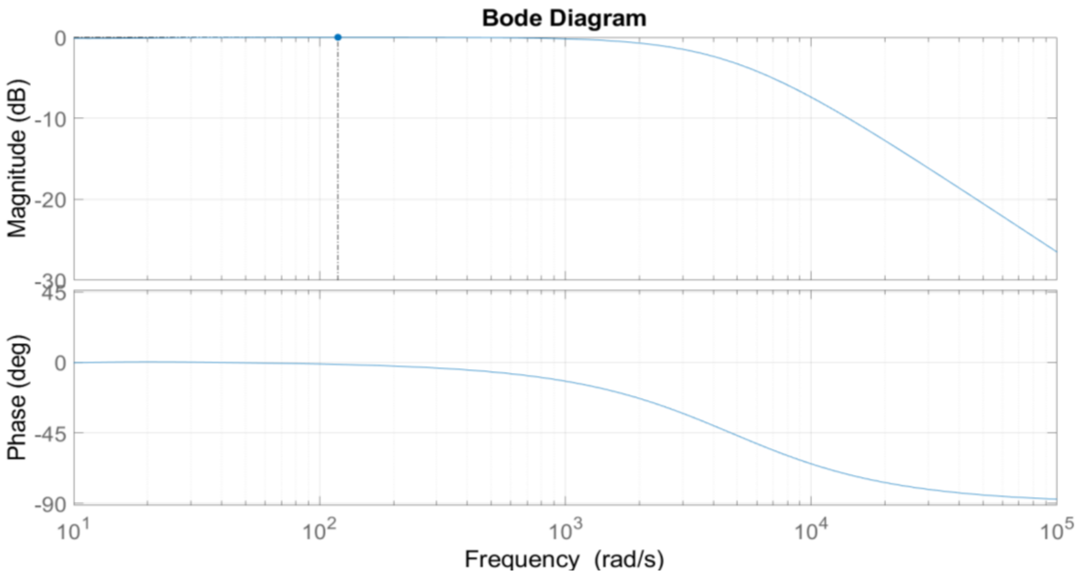

Figure 4. Bode response for the system

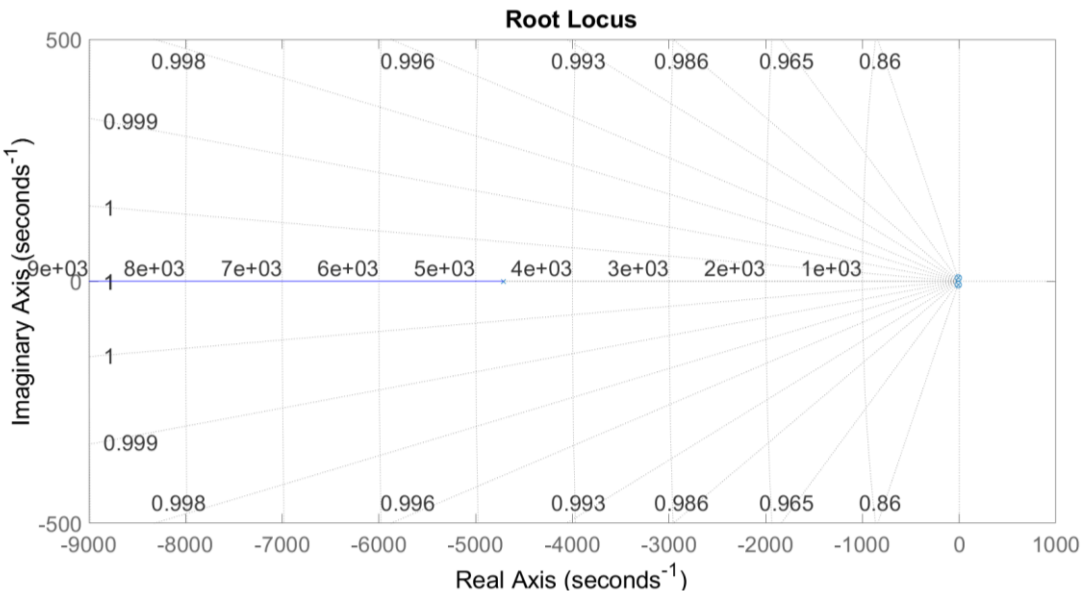

Figure 5. Root locus response for the system

Table 1. Controllers parameters and values of the performance criteria for the system

PID controller gain				Values of the performance criteria			
	k_p	k_i	k_d	Settling time (s)	Overshoot (%)	Rise time (s)	Peak time (s)
PID-1	1750824.3	5818890.3	47747.115	0.4577	4.0177	0.0041	0.0110
PID-2	9999900	7.4216e+07	5.0861e+05	7.9226e-04	0.2762	4.5876e-04	0.0015

4. SIMULATION STUDIES

The system performance in El-Centro and Northridge earthquake excitations has been investigated by applying a PID controller to the model in Figure 1. The efficiency of the determined PID parameters has been compared with the parameters obtained using the Matlab-Simulink Toolbox. Among the applied control states, PID-1 represents the controllers, which are the parameters determined by the Matlab-Simulink toolbox, and PID-2, the parameters obtained with the proposed method. In addition, the displacement and acceleration responses of the SDOF system have been examined as evaluation criteria. Figure 6 and Figure 7 show the displacement responses of the system, and Figure 8 and Figure 9 show the acceleration responses of the system. In the displacement responses of the El-Centro and Northridge earthquake excitations in Figure 6 and Figure 7, respectively, both controllers successfully suppressed the system responses. But the best performance has been obtained in the PID-2 control case.

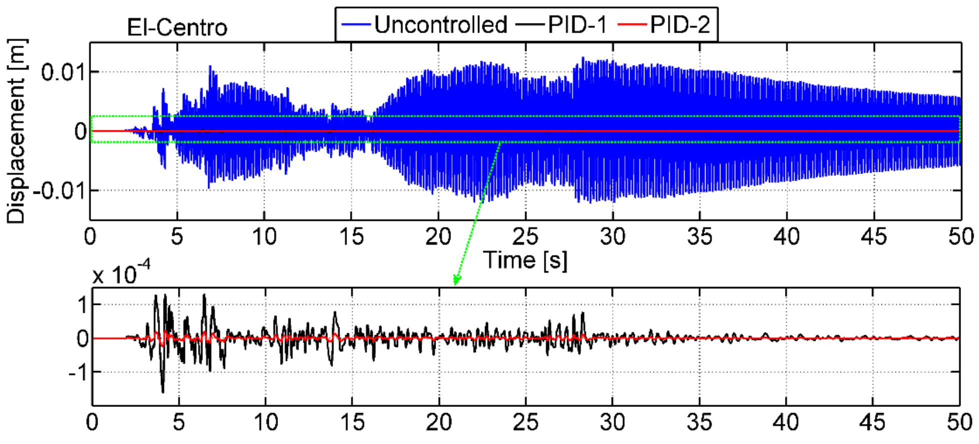

Figure 6. Displacement responses of the system under the El-Centro Earthquake

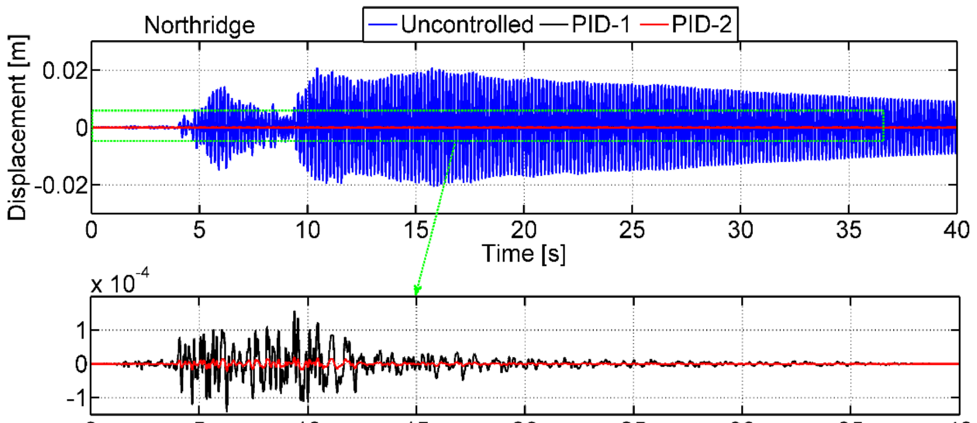

Figure 7. Displacement responses of the system under the Northridge Earthquake

In Figure 8 and Figure 9, acceleration responses in El-Centro and Northridge earthquake excitations, similar to displacements, both controllers successfully suppressed the system responses, and the best performance has been obtained in the PID-2 control condition.

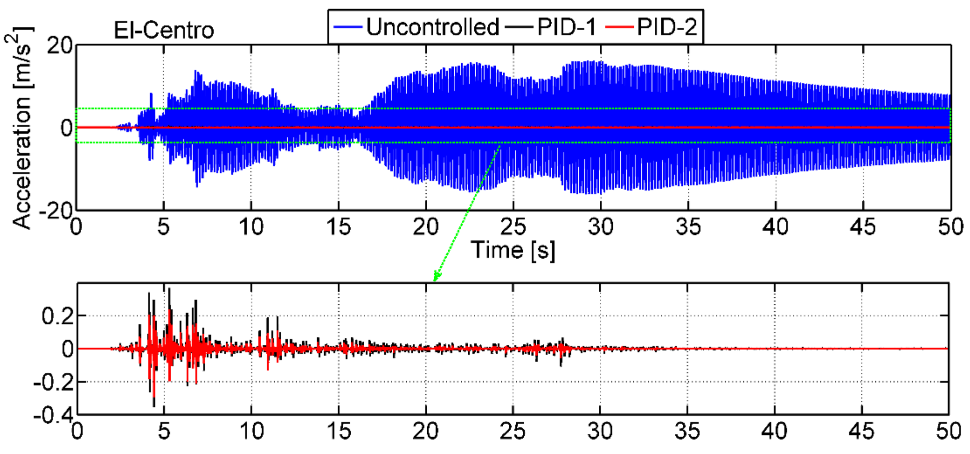

Figure 8. Acceleration responses of the system under the El-Centro Earthquake

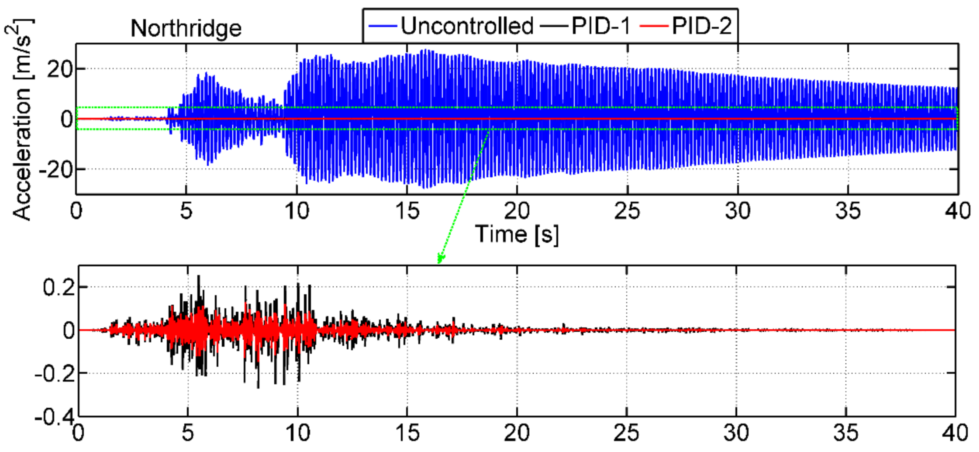

Figure 9. Acceleration responses of the system under the Northridge Earthquake

5. CONCLUSION

In this study, a design method is presented for optimum tuning of PID controller parameters. In the proposed method, other PID controller parameters are determined in response to the optimum proportional gain (k_p) setting in a stable loop so that the error rate of t_s and M_p is minimized. The effectiveness of this method, which has the advantages of not destroying parameters and not involving the complex mathematical formulation, is tested in an SDOF structural system under the influence of El-Centro and Northridge excitations. In addition, the performance of the PID controller (PID-2) obtained by the proposed method is compared with the controller (PID-1) obtained with the Matlab-Simulink PID toolbox.

The results reveal that PID-2 is not only effective at suppressing system responses but also outperforms PID-1.

REFERENCES

[1] M. Haskul and M. Kisa, Free-vibration analysis of cracked beam with constant width and linearly varying thickness, *Emerging Materials Research*, *vol. 11, no. 1, pp. 1-13, 2022.*

[2] M. Haskul and M. Kisa, Free vibration of the double tapered cracked beam. *Inverse Problems in Science and Engineering, vol 29, no.11, pp. 1537-1564, 2021.*

[3] K. J. Astrom and T. Hagglund, "Design of PI controllers based on non-convex optimization,'' *Automatica*, vol. 34, no. 5, pp. 585-601, 1998.

[4] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers,'' *Trans ASME*, vol. 64, pp. 759–768, 1942.

[5] G. H. Cohen and G. A. Coon, "Theoretical consideration of retarded control," *Trans ASME*, vol. 75, pp. 827–34, 1953.

[6] A. D. Paor and M. O'Malley, "Controllers of ziegler-nichols type for unstable process with time delay," *Int.J.Control*, vol. 49, pp. 1273–1284, 1989.

[7] V. Venkatashankar, and M. Chidambaram, "Design of P and PI controllers for unstable first-order plus time delay systems," *Int.J.Control*, vol. 60, pp. 137–144, 1994.

[8] K. J. Astrom and Hagglund, T. "The future of PID control," *Control Engineering Practice,* vol. 9, pp. 1163-1175, 2001.

[9] K. J. Manoj, and M. Chidambaram, "PID controller tuning for unstable systems by optimization method," *Chem.Engg.Comm.*, vol. 185, pp. 91–113, 2001.

[10] A. Visioli, "Optimal tuning of PID controllers for integral and unstable processes," *IEE Proc. in Control Theory Appln.*, vol. 148, pp. 148–180, 2001.

[11] C. Clement, and M. Chidambaram, "PID control of unstable FOPTD systems," *Chem. Eng. Commun.*, vol. 162, pp. 63–74, 1997.

[12] A. Turan, C. Onat and M. Sahin, "Active vibration suppression of a smart beam via PID controller designed through weighted geometric center method," *Proceedings of the 10th Ankara International Aerospace Conference*, METU, Ankara, Turkey, September, 2019.

[13] C. Onat, M. Daşkin, A. Turan, "Gain Scheduling PI Control of an electro-hydraulic actuator for active suspension system," 2nd International Conference On Computational Mathematics and Engineering Sciences (CMES-2017), İstanbul, Türkiye, May, 2017.

[14] R. Ali, T. H. Mohamed, Y. S. Qudaih and Y. Mitani, "A new load frequency control approach in an isolated small power systems using coefficient diagram method," *Int J Electr Power Energy Syst* vol. 56, pp. 110–116, 2014, https://doi.org/ 10.1016/j.ijepes.2013.11.002.

[15] A. Mittal, A. Kapoor and T. K. Saxena, "Adaptive tuning of PID controller for a nonlinear constant temperature water bath under set-point disturbances using GANFC," *J. Auto Syst. Eng.* vol. 7 pp. 143–63, 2013.

[16] A. Fereidouni, M. A. S. Masoum and M. Moghbel, "A new adaptive configuration of PID type fuzzy logic controller," *ISA Trans.* vol. 56, pp. 222–240, 2015, https://doi.org/ 10.1016/j.isatra.2014.11.010.

[17] X. Wu, G. Qin, H. Yu, S. Gao, L. Liu and Y. Xue, "Using improved chaotic and swarm to tune PID controller on cooperative adaptive cruise control," *Optik*, vol. 127 pp. 3445–3450, 2016, https://doi.org/10.1016/j.ijleo.2015.12.014.

[18] A. Moharam, M. A. El-Hosseini and H. A. Ali, "Design of optimal PID controller using hybrid differential evolution and particle swarm optimization with an aging leader and challengers," *Appl Soft Comput.* vol. 38 pp. 727–37, 2016, https://doi.org/ 10.1016/j.asoc.2015.10.041.

[19] F. N. Deniz, B. A. Alagoz and N. Tan, "PID Controller Design Based on Second Order Model Approximation by Using Stability Boundary Locus Fitting," *9th International Conference on Electrical and Electronics Engineering (ELECO),* 2015, 10.1109/ELECO.2015.7394585.

[20] P. Patel and S. Janardhanan, "Near optimal PID controller tuning: Interval arithmetic approach," *IFAC PapersOnLine* vol. 53, no. 1, pp. 246–251, 2020, 10.1016/j.ifacol.2020.06.042.

[21] R. Toscano, "A simple robust PI/PID controller design via numerical optimization approach," *Journal of Process Control*, vol. 15, pp. 81-88, 2005.

[22] H. Aggumus and R. Guclu, "Robust H_∞ control of STMDs used in structural systems by hardware in the loop simulation method," *In: Actuators. Multidisciplinary Digital Publishing Institute*, pp. 55, 2020.

[23] H. Aggumus, and S. Cetin, "Experimental investigation of semiactive robust control for structures with magnetorheological dampers," *Journal of Low Frequency Noise, Vibration and Active Control*, vol. 37, no. 2, pp. 216-234, 2018.

[24] R. Guclu and H. Yazici, "Seismic-vibration mitigation of a nonlinear structural system with an ATMD through a fuzzy PID controller," *Nonlinear Dynamics*, vol. 58, no. 3, pp. 553-564, 2009.

[25] R. Guclu, "Sliding mode and PID control of a structural system against earthquake," *Mathematical and Computer Modelling*, vol. 44, no. 1-2, pp. 210-217, 2006.

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:108-119 (2021)
Received: November 10, 2021 Accepted: December 1, 2021
JSCAI
ISSN: 2717-8226

Spectral Fletcher (CD) Algorithm for Solving Fuzzy Non-Linear Equations

[1]Mezher M. Abed , [2]Ufuk Öztürk , and [3]Hisham M. Khudhur

[1] Department Mathematics, Faculty of Science, University of Çankırı Karatekin, 18100 Çankırı, Turkey.
[2] Department Mathematics, Faculty of Science, University of Çankırı Karatekin, 18100 Çankırı, Turkey.
[3] Department Mathematics, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq.

Corresponding author: First A. Author (e-mail: hisham892020@uomosul.edu.iq).

ABSTRACT A conjugate gradient method is a powerful tool for solving large-scale miniaturization issues, with applications in arithmetic, chemistry, physics, engineering, medicine, and other fields. In this paper, we introduce a new spectral conjugate gradient algorithm, whose derivation is based on the Fletcher (CD) and Newton algorithms based on the solely coupling condition, which is introduced in this study. The significance of the research is in identifying a suitable algorithm. Because the Buckley and Qu methods are ineffectual in solving all types of ambiguous equations, and the conjugate gradient approach does not require a Hessian matrix (second partial derivatives of functions) in the solution, it is used to solve all types of ambiguous equations. The suggested method's descent property is demonstrated as long as the $\alpha_k$step size matches the strong Wolfe conditions. In many cases, numerical findings demonstrate that the novel technique is more efficient in solving nonlinear fuzzy equations than Fletcher (CD) algorithm.

KEYWORDS: algorithms, CG, Fletcher, fuzzy, numerical.

1. INTRODUCTION

Nonlinear equations can be solved using iterative approaches, such as

$$F(x) = 0 \tag{1}$$

It has received a lot of attention in recent years. The concept of fuzzy numbers, as well as the arithmetic operations that can be performed on them, were first proposed and researched by Zadeh [1]. One of the most popular applications for calculating fuzzy numbers is non-linear equations, the parameters of which are fully or partially represented by fuzzy numbers. [2]–[4]. Buckley and Qu's standard analytical procedures [5]–[8] are only suitable for the linear and quadratic cases of nonlinear equations. Cannot be used to solve equations like

I- $\mathrm{Q}y^3 + \mathrm{V}y^2 - \mathrm{L}y = \Theta,$
II- $\Theta e^y - \mathrm{Ky} = \Lambda,$
III- $\Phi \mathrm{y}\csc(y) + \Upsilon y = \Omega,$
IV- $\Psi y^5 - \Omega \cot(y) = \Phi.$

where $y, \mathrm{A}, \mathrm{B}, \mathrm{H}, \Theta, \mathrm{E}, \Lambda, \Upsilon, \Omega, \Psi,$ and Φ are fuzzy numbers. S. Abbasbandy and B. Asady employed Newton's method to solve a fuzzy nonlinear problem in 2004[9]. Amirah Ramli, Mohd Lazim Abdullah, and Mustafa Mamat used Quasi Newton's method to solve a fuzzy nonlinear problem in 2010 [10]. Due to the following disadvantages of the methodologies, this method is not particularly useful in practice: It necessitates the storage of the $n \times n$ matrix $[H_i]$, the computation of the elements of the matrix $[H_i]$ becomes extremely difficult and often impossible, the inversion of the matrix $[H_i]$ at each step, and the assessment of the amount $[H_i]^{-1}\nabla f_i$ at each step. The strategy is ineffective for challenges with a complex objective function with a large number of variables because of these flaws. The Steepest descent method for solving fuzzy nonlinear equations was developed by S. Abbasbandy and A. Jafarian [11]. which they published in 2004. However, because the sharpest descent direction is a local feature, this strategy is weak and inefficient in most applications. Hisham and Khalil created two conjugate gradient (CG) approach for solving fuzzy nonlinear equations in 202 [12], [13]. Mezher M. Abed, Ufuk ztürk, and Hisham M. Khudhur published Spectral CG Algorithm for Solving Fuzzy Non-linear Equations in 2022 [14]. Conjugate gradient methods have the drawback of being ineffective in some cases. Therefore, we need to use the new spectral gradient algorithm to find the roots of these equations, in this paper; We use one of the spectral conjugate gradient algorithms, because this algorithm is very efficient and fast in finding the roots of equations and also has global convergence. This paper is divided into six parts, the first part is a general introduction, previous studies, and the importance of the paper, the second part is the basics of arithmetic operations for fuzzy numbers, the third part is an explanation of the conjugate gradient algorithms, the fourth part is New Proposed Algorithm SCD-CG, the fifth part is the computational results and comparisons, and sixth part is the conclusions in addition to acknowledgments, and references.

2. PRELIMINARIES

We've gone through some basic definitions and arithmetic operations for fuzzy numbers in this section. For more information, we refer interested readers to [15].

Definition (2.1). A fuzzy number is defined as a set $j\colon \mathbb{R} \to I = [0,1]$ that meets the conditions listed below [16].

a. j denotes a semi-continuous upper boundary.
b. $j(x) = 0$ outside some range $[r, t]$.
c. there exist $p, q \in \mathbb{R}$ such that $r \le p \le q \le t$ and
   i. $j(x)$ is increasing monotonically on $[r, p]$.
   ii. $j(x)$ is decreasing monotonically on $[q, t]$.
   iii. $j(x) = 1,\ p \le x \le q$.

Definition (2.2). j: $\mathbb{R} \to I = [0,1]$ in parametric form

refer to the pair $(\underline{j}, \bar{j})$of $\underline{j}(\mu), \bar{j}(\mu),\ 0 \le \mu \le 1$

satisfying [16], [17],

(1) $\underline{j}(\mu)$ is a monotonically bounded growing left continuous function.

(2) $\bar{j}(\mu)$ is a monotonically bounded growing right continuous function.

(3) $\underline{j}(\mu) \le \bar{j}(\mu)$.

Definition (2.3). A classical Fuzzy number h refers to the Triangular number $j = (p, q, r)$ given as follows in equation (2)

$$j(x) = \begin{cases} \dfrac{(x-p)}{(r-p)}, & p \le x \le r \\ \dfrac{(x-q)}{(r-q)}, & r \le x \le q \end{cases} \tag{2}$$

with j(x) known as the membership function and $r \ne p, r \ne q$ [9]. This function can be written in its parameterized form as follows

$$\bar{j}(\mu) = q + (r-q)\mu$$
$$\underline{j}(\mu) = p + (r-p)\mu$$

Assume $TF(\mathbb{R})$ denotes the set of all trapezoidal fuzzy number. The following extension principle can be used to extend the scalar multiplication and addition operations to fuzzy numbers [9].

Let $= \left(\underline{j}(\mu), \bar{j}(\mu)\right)$, $k = (\underline{k}(\mu), \bar{k}(\mu))$ with $w > 0$, the addition $(j + k)$ and multiplication by scalar w are defined as

$$(\overline{j+k})(\mu) = \bar{j}(\mu) + \bar{k}(\mu)$$

$$\left(\underline{j+k}\right)(\mu) = \underline{j}(\mu) + \underline{k}(\mu)$$

$$(\overline{wj})(\mu) = w\bar{j}(\mu)$$

$$\left(\underline{wj}\right)(\mu) = w\underline{j}(\mu).$$

3. CONJUGATE GRADIENT (CG) ALGORITHMS

The non-linear conjugate gradient (CG) scheme has the form in equation (3)

$$x_{k+1} = x_k + \alpha_k d_k, \ \ k \geq 1 \tag{3}$$

Where x_1 is an initial paint, α_k is a step–length and α_k step-size that satisfy the standard Wolfe conditions in equation (4), and (5) [18], [19]

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k \tag{4}$$

$$d_k^T g(x_k + \alpha_k d_k) \geq \sigma d_k^T g_k \tag{5}$$

or strong Wolfe conditions in equation (6), and (7) [20]–[30],[31]

$$f(x_k + \alpha_k d_k) \leq f(x) + \delta \alpha_k g_k^T d_k \tag{6}$$

$$|d_k^T g(x_k + \alpha_k d_k)| \leq -\sigma d_k^T g_k \tag{7}$$

$$d_{k+1} = \begin{cases} -g_1, & k = 1 \\ -g_{k+1} + \beta_k d_k, & k \geq 1 \end{cases} \tag{8}$$

Different β_{k+1} will determine different CG methods. Some famous formula for β_{k+1}as follows:

The Fletcher and Reeves (FR) [32], Fletcher (CD) [33], Polak and Ribiere (PRP) [34], Hestenes-Stiefel (HS) [35], Dai-Yuan (DY) [36], Hisham- Khalil (KH) [12], and β is scalar.

$$\beta^{FR} = \frac{\| g_{k+1} \|^2}{\| g_k \|^2} \qquad \beta^{CD} = \frac{-\| g_{k+1} \|^2}{g_k^T d_k}$$

$$\beta^{HS} = \frac{g_{k+1}^T y_k}{y_k^T d_k} \qquad \beta^{PRP} = \frac{g_{k+1}^T y_k}{\| g_k \|^2}$$

$$\beta^{DY} = \frac{\| g_{k+1} \|^2}{y_k^T d_k} \qquad \beta^{KH} = \frac{\|g_{k+1}\|_1^2}{\|g_k\|_1^2},$$

Where $g_k = \nabla f(x_k)$, and let $y_k = g_{k+1} - g_k$.

4. NEW PROPOSED ALGORITHM SCD-CG

The search direction for the Fletcher (CD) conjugate gradient (CD-CG) method is obtained by $d_1 = -g_1$ and

$$d_{k+1} = -g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} d_k$$

The methods Fletcher and Reeves (FR) [32], Hisham- Khalil (KH) [12], Dai and Yuan (DY) [36], and Fletcher (CD) [33], have strong global convergence properties, but these methods have modest practical performance. While the methods Hestenes and Stiefel (HS) [35], and Polak and Ribiere (PRP) [34], are not always convergent, but they often have good computational properties see [37]. In order to obtain conjugate gradient methods with computational efficiency and good convergence properties, basically, the algorithms are found to avoid failure and to improve the performance of classical conjugated gradient algorithms. In order to accelerate the Fletcher (CD-CG) method, we use equation as follows

Let $\gamma_{k+1} = 1 + \mu_{k+1}$

Where γ_{k+1} and μ_{k+1} are two parameters, then

$$d_{k+1} = -\gamma_{k+1} g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} d_k$$

$$d_{k+1} = -(1 + \mu_{k+1}) g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} d_k \tag{9}$$

We incorporate the second-order information to the search direction in (9) by assuming, the direction in (9) is parallel to the Newton direction i.e

$$-G_{k+1}^{-1} g_{k+1} = -g_{k+1} - \mu_{k+1} g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} d_k \tag{10}$$

Where G_{k+1}^{-1} is the inverse Hessian matrix. Now suppose G_{k+1}^{-1} is symmetric $\left(G_{k+1}^{-1} = \left(G_{k+1}^{-1}\right)^T\right)$, positive definite and satisfies the Quasi-Newton condition i.e

$$G_{k+1}^{-1} y_k = s_k \tag{11}$$

Where $s_k = x_{k+1} - x_k$ multiply both sides of (10) by and considering G_{k+1}^{-1} be symmetric and positive definite, then

$$-\left(G_{k+1}^{-1} y_k\right)^T g_{k+1} = -y_k^T g_{k+1} - \mu_{k+1} y_k^T g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} y_k^T d_k$$

Use the relation given in (11) to get

$$-s_k^T g_{k+1} = -y_k^T g_{k+1} - \mu_{k+1} y_k^T g_{k+1} - \frac{\| g_{k+1} \|^2}{g_k^T d_k} y_k^T d_k$$

Divide both sides in the above equation by $y_k^T d_k$ then

$-\frac{s_k^T g_{k+1}}{y_k^T d_k} = -\beta_{k+1}^{HS} - \mu_{k+1}\beta_{k+1}^{HS} + \beta_{k+1}^{CD}$ (where $\beta_{k+1}^{HS} = \frac{g_{k+1}^T y_k}{y_k^T d_k}$, and $\beta_{k+1}^{CD} = -\frac{\|g_{k+1}\|^2}{g_k^T d_k}$)

Or

$$-\frac{s_k^T g_{k+1}}{y_k^T d_k} = -(1+\mu_{k+1})\beta_{k+1}^{HS} + \beta_{k+1}^{CD}$$

Or

$$(1+\mu_{k+1})\beta_{k+1}^{HS} = \beta_{k+1}^{CD} + \frac{s_k^T g_{k+1}}{y_k^T d_k}$$

$$\therefore \gamma_{k+1} = \frac{\beta_{k+1}^{CD}}{\beta_{k+1}^{HS}} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}} \tag{12}$$

Use this value for the γ_{k+1} in (9) then

$$d_{k+1} = -\gamma_{k+1} g_{k+1} + \beta_{k+1}^{CD} d_k \tag{13}$$

We call the algorithm defined in (12) and (13) as spectral Fletcher (SCD-CG) algorithm and we summarize it as the following algorithm SCD-CG.

Algorithm (SCD-CG):

step(1) : Initialization : select $x_1 \in R^n, \varepsilon > 0$ is a small positive real value and compute

$$d_1 = -g_1, \alpha_1 = 1/\|g_1\| \text{ and } k = 1$$

step(2) : Test for convergence: If $\|g_k\| \leq \varepsilon$ break x_k is optimal solution else go to step(3).

step(3) : Line search : calculateα_k satisfying the strong wolf conditions (6) (7) and up to date the variable $x_{k+1} = x_k + \alpha_k d_k$, calculate $f_{k+1}, g_{k+1}, y_k \; and \; s_k$.

step(4) : Direction calculation : calculate γ_{k+1} from (12), if $\gamma_{k+1} \geq 1$ or $\gamma_{k+1} \leq 0$ set $\gamma_{k+1} = 1$ and $d_{k+1} = -\gamma_{k+1} g_{k+1} + \beta_{k+1}^{CD} d_k$ then $d_{k+1} = -\gamma_{k+1} g_{k+1}$ else $d_{k+1} = d$ and $\alpha_{k+1} = \alpha_k * \|d_k\|/\|d_{k+1}\|, k = k+1$ go to step(2).

4.1 Descent property

In this part, we prove that our algorithm determines in equation (12) and (13) generates the descent direction for each iteration according to the following theorem.

Theorem:

consider the algorithm defined in equation (3) where d_k computed from (12) and (13). Assume that the step size α_k satisfies the strong Wolfe conditions (6) and (7). Then the search directions d_k generated by the SCD-CG algorithm are descent for all k provided $y_k^T g_{k+1} > 0$.

Proof

The prove is by indication, for k=1, $d_1 = -g_1 \to g_1^T d_1 < 0,$.

Now suppose $g_k^T d_k < 0$ or $g_k^T s_k < 0$ $s_k = \alpha_k d_k$ then for $k+1$ we have

$$d_{k+1} = -\left(\frac{\beta_{k+1}^{CD}}{\beta_{k+1}^{HS}} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1} + \beta_{k+1}^{CD} d_k$$

$$g_{k+1}^T d_{k+1} = -\left(\frac{\beta_{k+1}^{CD}}{\beta_{k+1}^{HS}} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - \frac{\alpha_k \|g_{k+1}\|^2}{\alpha_k g_k^T s_k} g_{k+1}^T s_k$$

$$g_{k+1}^T d_{k+1} = -\left(\frac{\beta_{k+1}^{CD}}{\beta_{k+1}^{HS}} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - \frac{\|g_{k+1}\|^2}{g_k^T s_k} g_{k+1}^T s_k$$

Divide both sides by $\frac{\|g_{k+1}\|^2}{g_k^T s_k}$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} = -\frac{g_k^T s_k}{\|g_{k+1}\|^2}\left(\frac{\beta_{k+1}^{CD}}{\beta_{k+1}^{HS}} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - g_{k+1}^T s_k$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\frac{g_k^T s_k}{\|g_{k+1}\|^2}\left(-\frac{y_k^T s_k}{\alpha_k g_{k+1}^T y_k}\frac{\alpha_k \|g_{k+1}\|^2}{g_k^T s_k} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - s_k^T g_{k+1}$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\frac{g_k^T s_k}{\|g_{k+1}\|^2}\left(-\frac{y_k^T s_k}{g_{k+1}^T y_k}\frac{\|g_{k+1}\|^2}{g_k^T s_k} + \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - s_k^T g_{k+1}$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\left(-\frac{y_k^T s_k}{g_{k+1}^T y_k} + \frac{g_k^T s_k}{\|g_{k+1}\|^2}\frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) g_{k+1}^T g_{k+1} - s_k^T g_{k+1}$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\left(-\frac{y_k^T s_k}{g_{k+1}^T y_k} g_{k+1}^T g_{k+1} + g_k^T s_k \frac{s_k^T g_{k+1}}{y_k^T g_{k+1}}\right) - s_k^T g_{k+1}$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\left(-\frac{y_k^T s_k}{g_{k+1}^T y_k} g_{k+1}^T g_{k+1} + g_k^T s_k \frac{s_k^T y_k}{y_k^T g_{k+1}}\right) - s_k^T y_k$$

$$\because s_k^T g_{k+1} = s_k^T g_{k+1} - s_k^T g_k + s_k^T g_k = s_k^T y_k + s_k^T g_k < s_k^T y_k$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\left(-\frac{y_k^T s_k}{g_{k+1}^T y_k} g_{k+1}^T g_{k+1} + g_k^T s_k \frac{s_k^T y_k}{y_k^T g_{k+1}}\right) - s_k^T y_k$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\frac{1}{y_k^T g_{k+1}}\left(-y_k^T s_k g_{k+1}^T g_{k+1} + g_k^T s_k s_k^T y_k\right) - s_k^T y_k$$

$$\frac{g_k^T s_k}{\|g_{k+1}\|^2} d_{k+1}^T g_{k+1} \leq -\frac{s_k^T y_k}{y_k^T g_{k+1}}\left(-g_{k+1}^T g_{k+1} + g_k^T s_k - y_k^T g_{k+1}\right)$$

$$d_{k+1}^T g_{k+1} \leq -\frac{s_k^T y_k}{y_k^T g_{k+1}}\left(-g_{k+1}^T g_{k+1} + g_k^T s_k - y_k^T g_{k+1}\right)\frac{\|g_{k+1}\|^2}{g_k^T s_k}$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} (-g_{k+1}^T g_{k+1} + g_k^T s_k - g_{k+1}^T g_{k+1} + g_k^T g_k) \frac{\|g_{k+1}\|^2}{g_k^T s_k}$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} (-2g_{k+1}^T g_{k+1} + g_k^T s_k + g_k^T g_k) \frac{\|g_{k+1}\|^2}{g_k^T s_k}$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \left(-\frac{2\left(g_{k+1}^T g_{k+1}\right)^2}{g_k^T s_k} + g_k^T s_k \frac{\|g_{k+1}\|^2}{g_k^T s_k} + g_k^T g_k \frac{\|g_{k+1}\|^2}{g_k^T s_k} \right)$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \left(-\frac{2\left(g_{k+1}^T g_{k+1}\right)^2}{g_k^T s_k} + \|g_{k+1}\|^2 + g_k^T g_k \frac{\|g_{k+1}\|^2}{g_k^T s_k} \right)$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \|g_{k+1}\|^2 \left(-\frac{2\|g_{k+1}\|^2}{g_k^T s_k} + 1 + \frac{g_k^T g_k}{g_k^T s_k} \right)$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \|g_{k+1}\|^2 \left(1 + \frac{g_k^T g_k}{g_k^T s_k} - \frac{2\|g_{k+1}\|^2}{g_k^T s_k} \right)$$

Use the Cuchy-Schuarz inequality then

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \|g_{k+1}\|^2 \left(1 + \frac{g_k^T g_k}{g_k^T s_k} - \frac{2\|g_{k+1}\|^2}{g_k^T s_k} \right)$$

$$= -\frac{s_k^T y_k}{y_k^T g_{k+1}} \|g_{k+1}\|^2 \left(1 + \frac{g_k^T g_k}{g_k^T s_k} - 2\beta_{k+1}^{CD} \right)$$

$$d_{k+1}^T g_{k+1} \le -\frac{s_k^T y_k}{y_k^T g_{k+1}} \|g_{k+1}\|^2 \left(1 + \frac{g_k^T g_k}{g_k^T s_k} - 2\beta_{k+1}^{CD} \right)$$

$s_k^T y_k > 0$ by Wolfe condition and $y_k^T g_{k+1} > 0$ by assumption

$\therefore d_{k+1}^T g_{k+1} < 0$

The proof is complete.

5. COMPARISONS AND COMPUTATIONAL RESULTS

This part presents the performance of Matlab 2021b implementation with hp laptop Ram 4GB, and hard 500GB of our new spectral conjugate gradient algorithm (SCD-CG) on a set of Fuzzy Nonlinear Equations taken from [7]. We compared the performance of this algorithm against the Fletcher algorithm (CD), and the algorithms were compared with the number of iterations, the optimal value of the function, and the optimal value of the variables as shown in Table (1). These algorithms are implemented with standard Wolfe conditions with $\rho = 0.1$ and $\sigma = 0.11$ where the initial step-size $\alpha = \frac{1}{\|g_k\|}$ and initial guess for other iterations i.e. $(k > 1)$ is $\alpha_k = \alpha_{k-1}\frac{\|d_{k-1}\|}{d_k}$. In the all cases the stopping criterion is $\|g_{k+1}\| \le 10^{-6}$ and maximum number of iteration is 2000. Our comparison includes the following.

1- It:- Number of Iterations
2- x-best:- optimal Variable
3- f-best:- optimal Function Value

Tables (1), show the details of the results for (SCD-CG) algorithms versus FR-CG algorithm. The numerical solutions were also plotted in Figs (1), (2), and (3).

Example 1: Consider the fuzzy nonlinear equation [7]

$(3,4,5)x^2 + (1,2,3)x = (1,2,3)$

Without any loss of generality, assume that x is positive, and then the parametric form of this equation is as follows:

$$\begin{cases}(3+r)\underline{x}^2(r) + (1+r)\underline{x}(r) - (1+r) = 0,\\ (5-r)\overline{x}^2(r) + (3-r)\overline{x}(r) - (3-r) = 0.\end{cases}$$

The above system needs initial values as follows. For $r = 1$

$$\begin{cases}4\underline{x}^2(1) + 2\underline{x}(1) - 2 = 0,\\ 4\overline{x}^2(1) + 2\overline{x}(1) - 2 = 0,\end{cases}$$

For $r = 0$

$$\begin{cases}3\underline{x}^2(0) + \underline{x}(0) - 1 = 0,\\ 5\overline{x}^2(0) + \overline{x}(0) - 3 = 0\end{cases}$$

With initial values

$x_0 = (\underline{x}(0), \underline{x}(1), \overline{x}(1), \overline{x}(0)) = (0.434, 0.5, 0.5, 0.681).$

Example 2: Consider the fuzzy nonlinear equation [7]

$(4,6,8)x^2 + (2,3,4)\,x - (8,12,16) = (5,6,7)$

Without any loss of generality, assume that x is positive, and then the parametric form of this equation is as follows:

$$\begin{cases}(4+2r)\underline{x}^2(r) + (2+r)\underline{x}(r) - (3+3r) = 0,\\ (8-2r)\overline{x}^2(r) + (4-r)\overline{x}(r) - (9-3r) = 0.\end{cases}$$

The above system needs initial values as follows. For $r = 1$

$$\begin{cases}6\underline{x}^2(1) + 3\underline{x}(1) - 6 = 0,\\ 6\overline{x}^2(1) + 3\overline{x}(1) - 6 = 0,\end{cases}$$

For $r = 0$

$$\begin{cases}4\underline{x}^2(0) + 2\underline{x}(0) - 3 = 0,\\ 8\overline{x}^2(0) + 4\overline{x}(0) - 9 = 0,\end{cases}$$

With initial values

$x_0 = (\underline{x}(0), \underline{x}(1), \overline{x}(1), \overline{x}(0)) = (0.651, 0.7808, 0.7808, 0.8397).$

Example 3: Consider the fuzzy nonlinear equation [7]

$(1,2,3)x^3 + (2,3,4)x^2 + (3,4,5) = (5,8,13)$

Without any loss of generality, assume that x is positive, and then the parametric form of this equation is as follows:

$$\begin{cases}(1+r)\underline{x}^3(r) + (2+r)\underline{x}^2(r) - (2+2r) = 0,\\ (3-r)\overline{x}^3(r) + (4-r)\overline{x}^2(r) - (8-4r) = 0.\end{cases}$$

The above system needs initial values as follows. For $r = 1$

$$\begin{cases}2\underline{x}^3(1) + 3\underline{x}^2(1) - 4 = 0,\\ 2\overline{x}^3(1) + 3\overline{x}^2(1) - 4 = 0,\end{cases}$$

For $r = 0$

$$\begin{cases}\underline{x}^3(0) + 2\underline{x}^2(0) - 2 = 0,\\ 3\overline{x}^3(0) + 4\overline{x}^2(0) - 8 = 0.\end{cases}$$

With initial values

$x_0 = (\underline{x}(0), \underline{x}(1), \overline{x}(1), \overline{x}(0)) = (0.76, 0.91, 0.91, 1.06)$.

Table (1) of numerical results for examples above

Examples	***CD ALGORITHM***			***SCD-CG ALGORITHM***		
	It	***x-best***	***f-best***	***It***	***x-best***	***f-best***
1	8	0.4343 0.5000 0.5000 0.5307	8.1709e-014	9	0.4343 0.5000 0.5000 0.5307	2.7933e-020
2	14	0.6514 0.7808 0.7808 0.8397	5.9506e-010	9	0.6514 0.7808 0.7808 0.8397	1.8868e-011
3	135	0.8393 0.9108 0.9108 1.0564	1.4978e-008	10	0.8393 0.9108 0.9108 1.0564	1.6416e-011

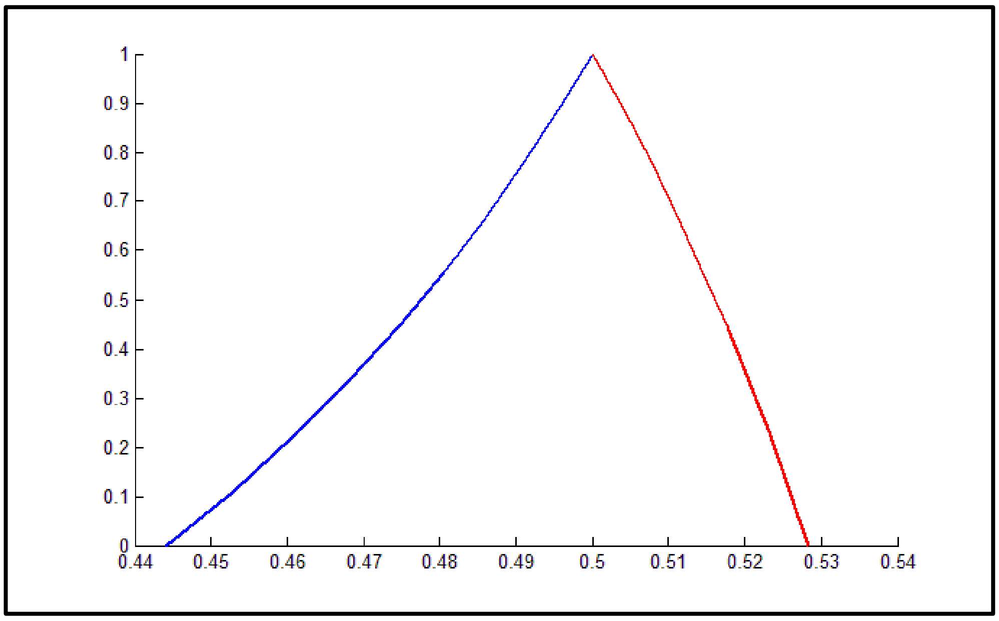

Figure 1. Drawing Solution to Example 1. using SCD-CG Algorithm

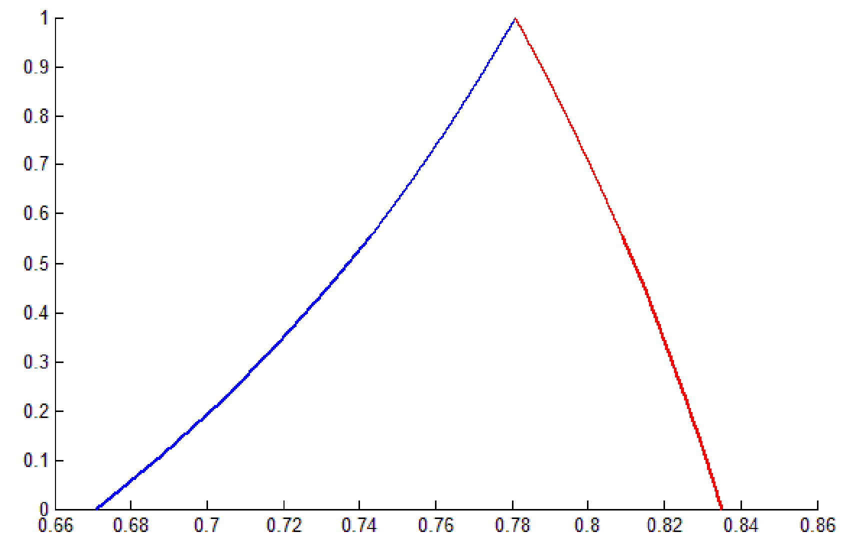

Figure 2: Drawing Solution to Example 2. using SCD-CG Algorithm

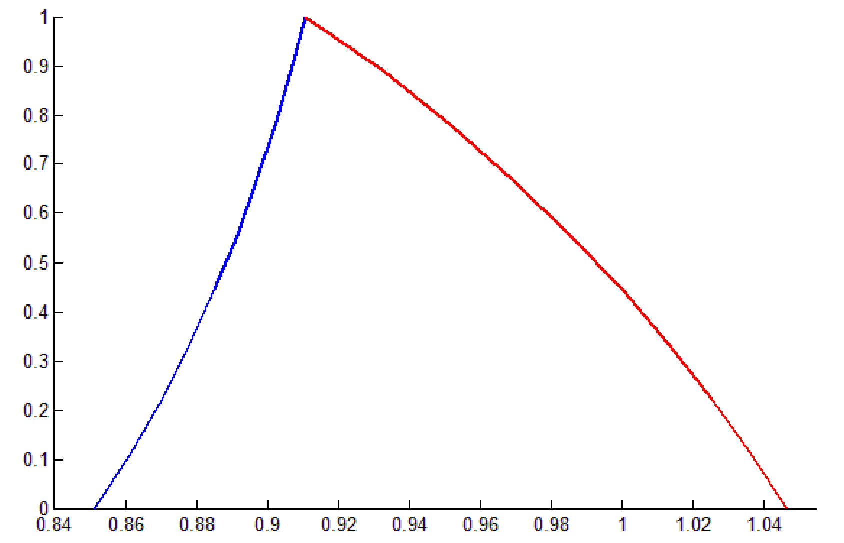

Figure 3: Drawing Solution to Example 3. using SCD-CG Algorithm

6. CONCLUSIONS

The main purpose of this article was to apply the spectral conjugate gradient algorithm that can be used to solve fuzzy nonlinear equations as an alternative to the usual analytical technique. The ambiguous nonlinear problem was transformed into a parametric formula and then solved using spectral conjugate gradient algorithms. The numerical results showed that the Spectral Fletcher algorithm (SCD-CG) performs very encouragingly in all the tested problems that have been solved, and this algorithm can also be used in other fields such as artificial neural networks, fuzzy neural networks, swarming algorithms, as well as in solving optimization problems.

ACKNOWLEDGMENTS

I would like to thank the College of Computer Science and Mathematics at Mosul University, and the Institute of Science and Technology in Çankiri Karatekin University for providing resources that have helped me increase the quality of my work.

REFERENCES

[1] L. A. Zadeh, “The concept of a linguistic variable and its application to approximate reasoning-III,” *Inf. Sci. (Ny).*, vol. 9, no. 1, pp. 43–80, 1975, doi: 10.1016/0020-0255(75)90017-1.

[2] R. Badard, “The law of large numbers for fuzzy processes and the estimation problem,” *Inf. Sci. (Ny).*, vol. 28, no. 3, 1982, doi: 10.1016/0020-0255(82)90046-9.

[3] P. Diamond, “Fuzzy least squares,” *Inf. Sci. (Ny).*, vol. 46, no. 3, pp. 141–157, 1988.

[4] M. Friedman, M. Ming, and A. Kandel, “Fuzzy linear systems,” *Fuzzy sets Syst.*, vol. 96, no. 2, pp. 201–209, 1998.

[5] J. J. Buckley and Y. Qu, “Solving linear and quadratic fuzzy equations,” *Fuzzy Sets Syst.*, vol. 38, no. 1, pp. 43–59, 1990, doi: 10.1016/0165-0114(90)90099-R.

[6] J. J. Buckley and Y. Qu, “On using α-cuts to evaluate fuzzy equations,” *Fuzzy Sets and Systems*, vol. 43, no. 1. Elsevier, p. 125, 1991, doi: 10.1016/0165-0114(91)90026-M.

[7] J. J. Buckley and Y. Qu, “Solving fuzzy equations: A new solution concept,” *Fuzzy Sets Syst.*, vol. 39, no. 3, pp. 291–301, 1991, doi: 10.1016/0165-0114(91)90099-C.

[8] J. J. Buckley and Y. Qu, “Solving systems of linear fuzzy equations,” *Fuzzy Sets Syst.*, vol. 43, no. 1, pp. 33–43, 1991, doi: 10.1016/0165-0114(91)90019-M.

[9] S. Abbasbandy and B. Asady, “Newton’s method for solving fuzzy nonlinear equations,” *Appl. Math. Comput.*, vol. 159, no. 2, pp. 349–356, Jul. 2004, doi: 10.1016/j.amc.2003.10.048.

[10] M. Mamat, A. Ramli, and M. L. Abdullah, “Broyden’s method for solving fuzzy nonlinear equations,” *Adv. Fuzzy Syst.*, 2010, doi: 10.1155/2010/763270.

[11] S. Abbasbandy and A. Jafarian, “Steepest descent method for solving fuzzy nonlinear equations,” *Appl. Math. Comput.*, vol. 174, no. 1, pp. 669–675, Jul. 2006, doi: 10.1016/j.amc.2005.04.092.

[12] H. M. Khudhur and K. K. Abbo, “A New Type of Conjugate Gradient Technique for Solving Fuzzy Nonlinear Algebraic Equations,” in *Journal of Physics: Conference Series*, 2021, vol. 1879, no. 2, doi: 10.1088/1742-6596/1879/2/022111.

[13] H. M. Khudhur and K. K. Abbo, “New hybrid of Conjugate Gradient Technique for Solving Fuzzy Nonlinear Equations,” *J. Soft Comput. Artif. Intell.*, vol. 2, no. 1, pp. 1–8, 2021.

[14] M. M. Abed, U. Öztürk, and H. Khudhur, “Spectral CG Algorithm for Solving Fuzzy Non-linear Equations,” *Iraqi J. Comput. Sci. Math.*, vol. 3, no. 1, 2022.

[15] L. A. Zadeh, “Fuzzy sets,” *Inf. Control*, vol. 8, no. 3, pp. 338–353, 1965.

[16] D. J. Dubois, *Fuzzy sets and systems: theory and applications*, vol. 144. Academic press, 1980.

[17] H. Zimmermann, “Fuzzy set theory and its applications,” *Int. Ser. Manag.*, 1991.

[18] B. Hassan, H. Jabbar, and A. Al-Bayati, “A new class of nonlinear conjugate gradient method for solving unconstrained minimization problems,” 2019, doi: 10.1109/ICCISTA.2019.8830657.

[19] B. A. Hassan, “A new type of quasi-newton updating formulas based on the new quasi-newton equation,” *Numer. Algebr. Control Optim.*, vol. 10, no. 2, pp. 227–235, 2020, doi: 10.3934/naco.2019049.

[20] WOLFE P, "CONVERGENCE CONDITIONS FOR ASCENT METHODS," *SIAM Rev.*, vol. 11, no. 2, pp. 226–235, 1969, doi: 10.1137/1011036.

[21] H. M. Azzam, H. N. Jabbar, and K. K. Abo, "Four–Term Conjugate Gradient (CG) Method Based on Pure Conjugacy Condition for Unconstrained Optimization," *Kirkuk Univ. Journal-Scientific Stud.*, vol. 13, no. 2, pp. 101–113, 2018, doi: 10.32894/kujss.2018.145720.

[22] Y. A. Laylani, K. K. Abbo, and H. M. Khudhur, "Training feed forward neural network with modified Fletcher-Reeves method," *J. Multidiscip. Model. Optim.*, vol. 1, no. 1, pp. 14–22, 2018, [Online]. Available: http://dergipark.gov.tr/jmmo/issue/38716/392124#article_cite.

[23] A. S. Ahmed, "Optimization Methods For Learning Artificial Neural Networks," University of Mosul, 2018.

[24] A. S. Ahmed, H. M. Khudhur, and M. S. Najmuldeen, "A new parameter in three-term conjugate gradient algorithms for unconstrained optimization," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 23, no. 1, 2021, doi: 10.11591/ijeecs.v23.i1.pp338-344.

[25] H. M. Khudhur, "Numerical and analytical study of some descent algorithms to solve unconstrained Optimization problems," University of Mosul, 2015.

[26] K. K. Abbo and H. M. Khudhur, "New A hybrid conjugate gradient Fletcher-Reeves and Polak-Ribiere algorithm for unconstrained optimization," *Tikrit J. Pure Sci.*, vol. 21, no. 1, pp. 124–129, 2015.

[27] K. K. Abbo and H. M. Khudhur, "New A hybrid Hestenes-Stiefel and Dai-Yuan conjugate gradient algorithms for unconstrained optimization," *Tikrit J. Pure Sci.*, vol. 21, no. 1, pp. 118–123, 2015.

[28] K. K. ABBO, Y. A. Laylani, and H. M. Khudhur, "A NEW SPECTRAL CONJUGATE GRADIENT ALGORITHM FOR UNCONSTRAINED OPTIMIZATION," *Int. J. Math. Comput. Appl. Res.*, vol. 8, pp. 1–9, 2018.

[29] K. K. Abbo, Y. A. Laylani, and H. M. Khudhur, "Proposed new Scaled conjugate gradient algorithm for Unconstrained Optimization," *Int. J. Enhanc. Res. Sci. Technol. Eng.*, vol. 5, no. 7, 2016.

[30] Z. M. Abdullah, M. Hameed, M. K. Hisham, and M. A. Khaleel, "Modified new conjugate gradient method for Unconstrained Optimization," *Tikrit J. Pure Sci.*, vol. 24, no. 5, pp. 86–90, 2019.

[31] H. M. Khudhur and K. K. Abbo, "A New Conjugate Gradient Method for Learning Fuzzy Neural Networks," *J. Multidiscip. Model. Optim.*, vol. 3, no. 2, pp. 57–69, 2020.

[32] R. Fletcher and C. M. Reeves, "Function minimization by conjugate gradients," *Comput. J.*, vol. 7, no. 2, pp. 149–154, 1964, doi: 10.1093/comjnl/7.2.149.

[33] C. Witzgall and R. Fletcher, "Practical Methods of Optimization.," *Math. Comput.*, vol. 53, no. 188, p. 768, 1989, doi: 10.2307/2008742.

[34] E. Polak and G. Ribiere, "Note sur la convergence de méthodes de directions conjuguées," *Rev. française d'informatique Rech. opérationnelle. Série rouge*, vol. 3, no. 16, pp. 35–43, 1969, doi: 10.1051/m2an/196903r100351.

[35] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Natl. Bur. Stand. (1934).*, vol. 49, no. 6, p. 409, 1952, doi: 10.6028/jres.049.044.

[36] Y. H. Dai and Y. Yuan, "A nonlinear conjugate gradient method with a strong global convergence property," *SIAM J. Optim.*, vol. 10, no. 1, pp. 177–182, 1999, doi: 10.1137/S1052623497318992.

[37] Z.-J. Shi and J. Guo, "A new algorithm of nonlinear conjugate gradient method with strong convergence," *Comput. Appl. Math.*, vol. 27, pp. 93–106, 2008.

Journal of Soft Computing and Artificial Intelligence
Research Article 2(2): pp:120-131 (2021)
Received: December 7, 2021 Accepted: December 15, 2021
JSCAI
ISSN: 2717-8226

Forecasting Turkish Lira (TRY)/US Dollar (USD) Interest Exchange Rates Using Machine Learning Methodologies

Mahmut Dirik
Sirnak University, Department of Computer Engineering, Turkey
Corresponding author: First A. Author (e-mail: mhmd.dirik@gmail.com).

ABSTRACT Machine learning algorithms have become increasingly popular in recent years for analyzing financial data and predicting the exchange rate system. The aim of this paper was to construct an investment appreciation rate estimation model based on machine learning by estimating the Turkish lira/US dollar exchange rate. The forecasting model was developed using foreign exchange market data, namely the exchange rates in TL and USD at specific periods. The proposed model was estimated using machine learning methods such as Multilayer Perceptron (MLP), Linear Regression (LR), Support Vector Machine (SVM), Random Forest (RF), and Local Weighted Learning (LWL). The model's validity was established using TRY interest rates and the USD exchange rate. The data were analyzed using mean absolute error (MAE), directional accuracy (DA), mean square error (MSE), and root mean square error (RMSE). These metric results show that the proposed model is suitable for both prediction and investment data.

KEYWORDS: Machine Learning, Exchange Rate Prediction, Regression, MLP, SVM, RF, and LWL.

1. INTRODUCTION

The exchange rate is one of the most important economic indicators of a country's economy. When it comes to making financial decisions, investors and businesses pay close attention to exchange rates. International investors pay close attention to the exchange rate when making investment decisions because it has the potential to cause financial damage. Many factors, such as the international trade balance, central bank fiscal policies, and exchange rates, affect the value of foreign currency rates. Cryptographic forecasting is very important in the economic field and has been studied for quite some time [1]. In this capacity, forex market forecasts are critical to investors and businesses [2] [3]. Financial data analysis can be divided into two distinct categories: fundamental and technical analysis. Fundamental analysts use macroeconomic difficulties to forecast problems in financial markets, while models based on historical data focus only on past data. Fundamental macroeconomic analysis and sound technical financial analysis are two commonly used types of financial analysis. While the former focuses on the crucial market characteristics, the latter is much more extensive and detailed [4]. Technical analysis assumes that the behavior of a financial market reflects the emotions of investors. In sound technical financial research, various techniques are used to determine how an investor's sentiment evolves over time. For example, "moving averages" represent trends within a particular investment. On the other hand, macroeconomic fundamental analysis is based on the idea that changes in the macroeconomic environment will lead to changes [5]. Pricing methods based on model characteristics are notoriously difficult to forecast. As a result, investors know that such investments require some risk while they actively seek the maximum potential return. As research on the potential downsides of private equity, venture capital, and other illiquid investments has shown, risk-averse investors are better served by investing in more liquid investments. These investors may be looking for the highest potential return on their investments. Because of the unpredictability of these investments, some believe that the risk is always greater than the potential gain [6]. According to the technical analysis, well-known statistical models such as the autoregressive moving average and generalized autoregressive conditional change in variance (GARCH) [7] [8] do not satisfy the nonlinearity of time series data. They also do not account for recurrent trends in time series data. Recurrent neural architecture models can represent recurrent patterns in time series data by using time series based gradient descent learning. Several advanced machine learning technologies (ML) [9-12] have recently led to better results in predicting time series. Artificial neural networks are one of them (ANN). For financial time series data, ANN has proven to be an excellent choice [10]. The success of ANNs is largely attributed to the fact that they find a nonlinear relationship without prior understanding of the information and have the ability to train and predict themselves. Machine learning has changed the game [11]. ANN is essentially a network of nodes organized into layers and connected by links with associated weights. To train the network, some training data is fed into the network. This training helps in selecting a decision function from a set of functions represented by the ANN structure. This decision function can be determined by assigning sufficient weights to the network. The error is reduced if the weights are chosen correctly [12]. Backpropagation Neural Network is a typical approach for error reduction, where the error is fed backwards to change the weights, resulting in error minimization [13].With good features and a suitable network model, such as a forward or backward model, high accuracy can be achieved. The aim of this project is to develop a machine learning model to predict the exchange rate of the Turkish Lira to the US Dollar using machine learning techniques. The main purpose is to develop an intelligent model that incorporates critical variables such as the TRY crude interest rate and the USD price index. The exchange rate is particularly susceptible to fluctuations in investor sentiment as it reacts to macroeconomic developments. Therefore, we have adopted an ML approach that is immune to sensitivity changes. Our data set includes nonlinear time-dependent data sets. The years 2010 to 2021 are covered by these datasets. The dataset has ten matched columns: Date, one-month Turkish lira interest rates, three-month Turkish lira interest rates, six-month Turkish lira interest rates, one-year Turkish lira interest rates, one-month USD interest rates, three-month Turkish lira interest rates, six-month USD interest rates, and one-year USD interest rates are all included [14].

The remainder of the article is organized as follows: Section 2 discusses the proposed methodology, algorithm, and procedures; Section 3 explains the experimental work and results; and Section 4 discusses the conclusion.

2. PROPOSED METHODOLOGY

Due to its ability to handle nonlinearity and adapt to noise, ANN has been widely used as an alternative for autonomous prediction in recent years [1] [15]. ANN -based forecasting has been offered as a very effective method for financial forecasting. The main objective of the study is to develop a forecasting model that can predict exchange rate fluctuations. The paper proposes the use of ML algorithms-based time series forecasting models. After studying the data and comparing it with many other machine learning techniques, we found that LR is the best acceptable model for the forecasting problem. The dataset containing the interest rates between the Turkish lira and the dollar in certain time intervals and in certain periods, as well as the buying and selling information of dollars, is used in this model. Figure 1 summarizes the proposed approach to predict the USD-TRY conversion rate and interest rates between 2010 and 2021. The software Weka [16] was used for the analysis carried out here.

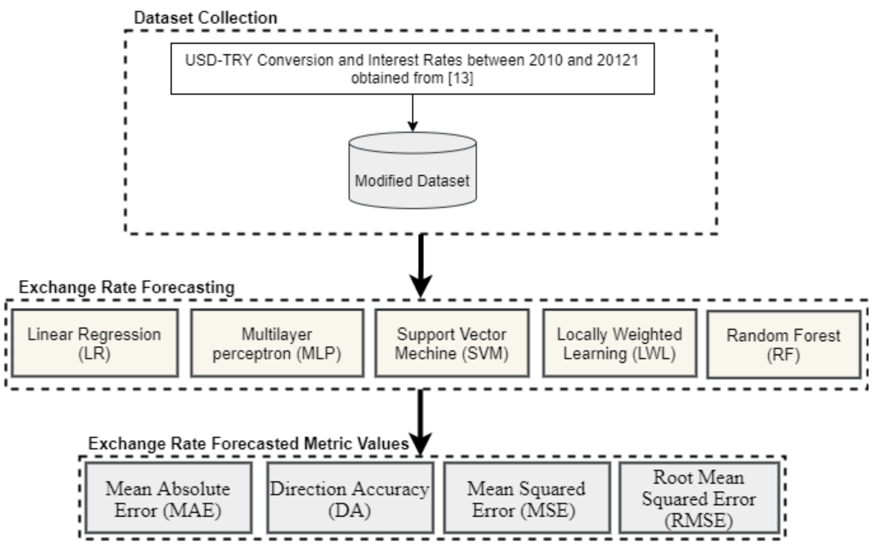
Dataset Collection
USD-TRY Conversion and Interest Rates between 2010 and 20121 obtained from [13]
Modified Dataset
Exchange Rate Forecasting
Linear Regression (LR)
Multilayer perceptron (MLP)
Support Vector Mechine (SVM)
Locally Weighted Learning (LWL)
Random Forest (RF)
Exchange Rate Forecasted Metric Values
Mean Absolute Error (MAE)
Direction Accuracy (DA)
Mean Squared Error (MSE)
Root Mean Squared Error (RMSE)

Figure 1. The overview of the proposed methodology

In this paper, ML techniques are used to estimate the exchange rate between the Turkish lira and the US dollar using time series analysis, focusing on exchange rate estimates. Estimation is a statistical technique for predicting future values in a time series using past data. The purpose of estimation is to offer values for future data that are uncertain. These projections and estimates can then be used to prepare for the future, make forecasts, and create budgets for future revenues and expenses. The proposed solution is based on a machine learning model that forecasts the future interest rate of the currency using data from [14]. The input data consists of the changes in the interest rates for the USD and the Turkish lira over the last 1, 3, 6, and 12 months. The downward forecast is based on the change in the buying and selling rates of the dollar.

2.1 Dataset

The data used here is from the Kaggle website [14]. From this resource, changes and details about the data can be obtained. It is a dollar buy/sell dataset that illustrates how individuals respond to changing circumstances as a function of TL/dollar interest rates. The data used ranges from July 2010 to July 2021 and consists of 134 rows and 11 columns (characteristics). These include the following: 1-month Turkish lira interest rates, 3-month Turkish lira interest rates, 6-month Turkish lira interest rates and 1-year Turkish lira interest rates, 1-month USD interest rates, 3-month USD interest rates, 6-USD interest rates and 1-year USD interest rates. It is composed of the USD/TRY buy and sell conversion ratios [14].

2.2 Random Forest

Random forests are collections of tree predictors where each tree is determined by the values of a uniformly distributed random vector. As the number of trees in a forest increases, the generalization error approaches a threshold. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and their relationship to each other. Random forests are an efficient tool for prediction. They are not subject to overfitting by the law of large numbers. Because of their moderate predictability, they are accurate classifiers and regressors. In addition, the framework provides information about the predictive ability of the random forest, such as the strength of each predictor and their correlations. By using out-of-bag estimates, one can obtain actual numbers for strength and correlation that would otherwise have remained speculative [17]. It is calculated as in Eq. 1.

$$RF = E_{X,Y}(Y - h(X)^2 \quad (1)$$

Random forests are generated for regression by growing trees based on a random vector such that the tree predictor h(x,) contains numerical values rather than class labels. The output values are numerical, and we assume that the training set is randomly selected from the distribution of the random vector Y, X. The mean squared generalization error for each numerical predictor h(x) is computed by averaging the trees h (x, k) over k.

2.3 Linear Regression

Linear regression is a modeling and analysis tool for many different types of interactions between two or more variables. This variable may be the price of a stock in the financial market, the growth of a biological species, or the possibility of the discovery of a gravitational wave. A linear regression model describes the relationship between one or more independent variables, X, and a dependent variable, y. The dependent variable is also called the response variable. Continuous predictor variables are also called covariates, and categorical predictor variables are called factors. The design matrix is usually referred to as the X matrix of observations on the predictor variables [9].

2.4 Multi-layer Perceptron

A multilayer perceptron (MLP) is a variant of a feedforward neural network. As shown in Figure 2, this network has an input layer, an output layer, and a hidden layer. As in a feedforward network, data in an MLP is passed from the input layer to the output layer. The MLP neurons are trained using the backpropagation learning method. MLPs are supposed to be able to estimate any continuous function and solve problems that are linearly inseparable [18][19].

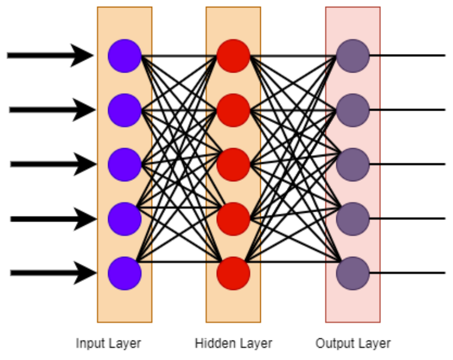

Figure 2. An MLP with a single hidden layer is shown schematically.

During the training phase, neurons in each layer are connected to neurons in the next layer by a weighted connection. In the buried layer, activation functions are used to accurately evaluate the features of the input data. Backpropagation training is a method for solving a variety of estimation problems using supervised learning. The following is an example of the output of an ANN layer.

$$a^i = f(\sum_{j=1}^{N} w_{ij}x_j + b^i) \tag{2}$$

Where w_{ij} and b denote the weights and bias values, respectively, N denotes the number of input neurons, and f denotes the activation function. More information about the MLP technique can be found here [20].

2.5 Support Vector Machine

Support Vector Machine (SVM) is one of the most widely used supervised learning algorithms in machine learning for classification and regression applications. Vladimir Vapnik [21] and colleagues first proposed SVM analysis in 1992, and it is a well-known machine learning technique for classification and regression. The goal of SVM is to find the optimal line or decision boundary for classifying the n-dimensional space, so that more data points can be classified easily in the future. SVM selects endpoints or vectors that help in forming the hyperplane. These extreme states are called support vectors, and the technology is called support vector machines. Since it is based on kernel features, SVM regression is a non-parametric approach. See [22] for more details.

2.6 Locally Weighted Learning

Locally weighted regression (LWR) is a nonparametric, memory-based regression technique that performs regression around a point of interest using only "local" training data. The techniques monitor and analyze the training data to make predictions. It is a method of fitting all training data to the domain around a query sample. Processing of the training data is often postponed until the target value of a query sample needs to be determined, as LWR is a type of lazy learning [23] [24].

2.7 Evaluation Metrics

Numerous metrics have been used to measure the efficiency of learning algorithms used to effectively predict the direction of the forex market. We evaluated the data in this study using four metrics: MAE, DA, RMSE, and MSE. These criteria can be used to evaluate investment data in terms of the performance of the algorithms used to estimate TL and USD exchange rates in the proposed model.

2.7.1 Mean Absolute Error (MAE)

Mean absolute error (MAE) is the most important indicator for measuring predictive accuracy. Unlike relative error, absolute error is the difference between expected and actual values. The MAE of a continuous variable is used to evaluate its accuracy. The mean absolute error of a prediction indicates the magnitude of the error that, on average, can be expected in the prediction. The mean absolute error estimates the average magnitude of error in a collection of predictions regardless of their direction. The MAE of a sample is defined as the average of the absolute differences between predicted and observed values, with all individual deviations weighted equally. The MAE and the RMSE can be used to characterize the model prediction error with respect to the target variable [25] [26]. This error is the average estimated error without considering the directions of the predicted values. Each estimated difference has the same weight and can be determined in the following way.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3}$$

Absolute errors are defined as "$|y_i - \hat{y}_i|$" and are the sum symbol.

2.7.2 Directional Accuracy (DA)

Measures of directional accuracy (DA) include both realized and magnitudes of directional changes in exchange rates. These metrics are robust to outliers and provide an economically interpretable loss/gain functional framework in a critical decision-theoretic environment for traders and investors.

DA is a metric that measures the predictive accuracy of a forecasting method. It compares the expected direction (up or down) with the actual direction. The following formula is used to define DA [27] [28].

$$DA = \frac{1}{N}\sum_{t} 1_{sign\,(A_t - A_{t-1})} = \\ = sign(F_t - A_{t-1}) \tag{4}$$

Where A_t is the current value at time t and F_t is the predicted value at time t. N is the number of prediction points. Sign is a function.

2.7.3 Mean Squared Error (MSE)

The mean squared error (MSE) can be used to measure the degree of fit between a regression curve and a collection of points. The MSE statistic is used to quantify the uncertainty associated with statistical models. It is defined as the difference in mean squared values between the observed and expected values. As the accuracy of the model decreases, so does its value. For example, the mean squared error reflects the mean squared residual of the regression [29]. The MSE is calculated using the following formula.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{5}$$

Where, y_i is the i-th observed value. $\hat{y}$ is the corresponding predicted value. N is the number of observations.

2.7.4 Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) is commonly used to quantify the discrepancy between what a model predicts and what is actually observed in the environment under study. The RMSE converts these individual variances into a single measure of predictive capacity, the residuals. The RMSE is a statistic that indicates the magnitude of the difference between two sets of data. In other words, the root mean square error is used to determine the difference between a predicted outcome and an actual outcome. You can use this metric to determine the average accuracy of your predictions [28] [29]. Calculate it using the following equation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{6}$$

In this case, y_i represents the actual data, while $\hat{y}_i$ represents the predicted data. N is the number of observations.

3. EXPERIMENTAL WORKS AND RESULTS

The data used in these experiments cover the period from July 2010 to July 2021 and include historical interest rates and USD/TRY conversion rates. Using the data from the experiments conducted in our research, we performed a financial sensitivity analysis [14]. Based on the performance of the algorithms used to analyze the data, various metric calculation results were evaluated and presented in the form of a table (Table 1). In this study, all the data were analyzed as training data and the results were compared in this way. 1, 3, 6 months, 1-year Turkish lira and USD interest rates as well as USD/TRY purchase conversion rate and USD/TRY sale conversion rate consist of the data used in the proposed model and calculated periodically as time series. The table shows the evaluation indicators for the models that perform best.

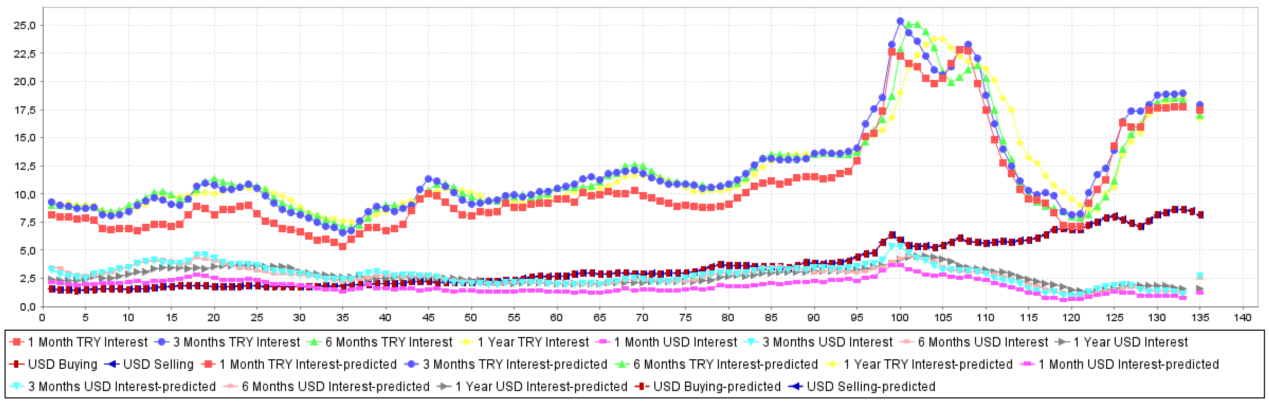

Figure 3. Future forecast for: 1 Month TRY Interest, 3 Month TRY Interest, 6 Month TRY Interest, 1 Year TRY Interest, 1 Month USD Interest, 3 Month USD Interest, 6 Month USD Interest, 1 Year USD Interest, USD Buying, USD Selling

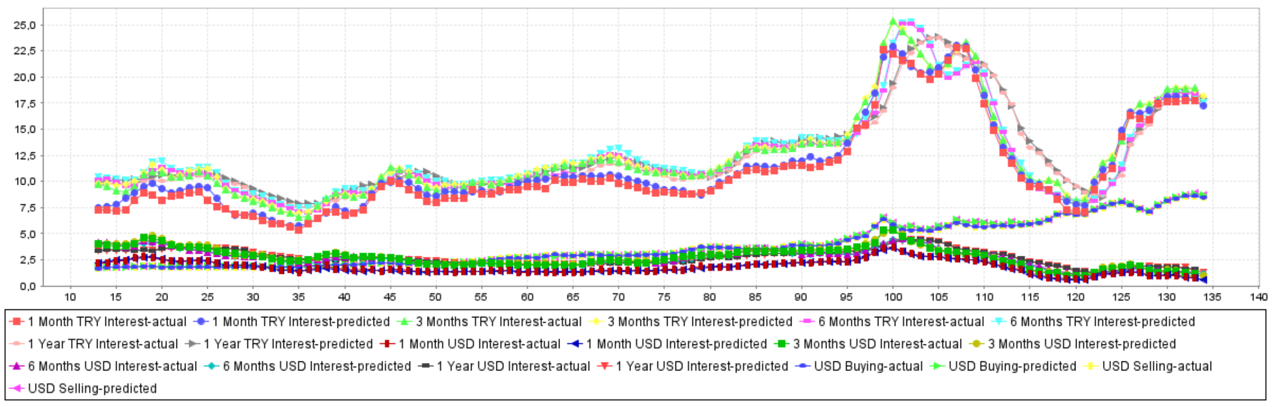

Figure 4. 1 step ahead predictions for: 1 Month TRY Interest, 3 Month TRY Interest, 6 Month TRY Interest, 1 Year TRY Interest, 1 Month USD Interest, 3 Month USD Interest, 6 Month USD Interest, 1 Year USD Interest, USD Buying, USD Selling

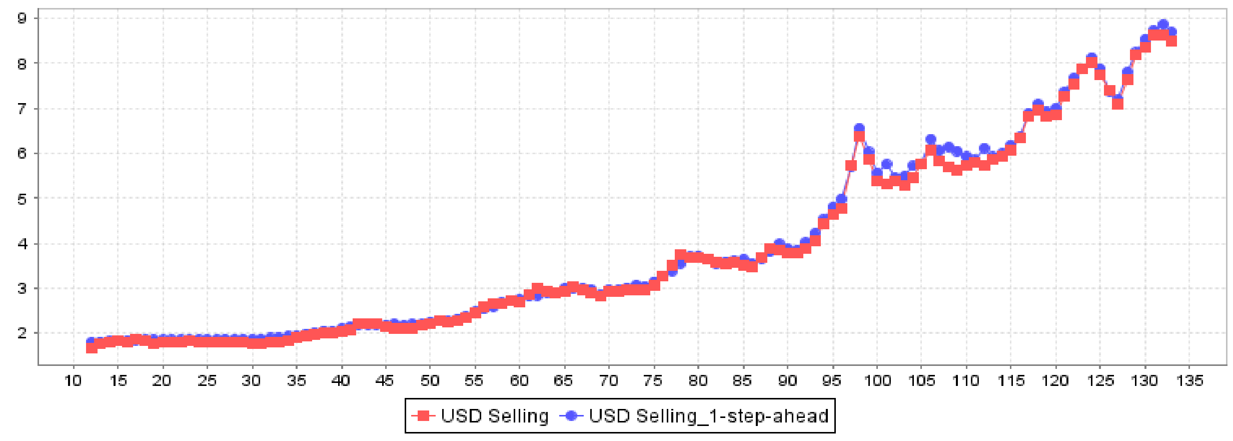

Figure 5. 1step-ahead prediction for USD Selling

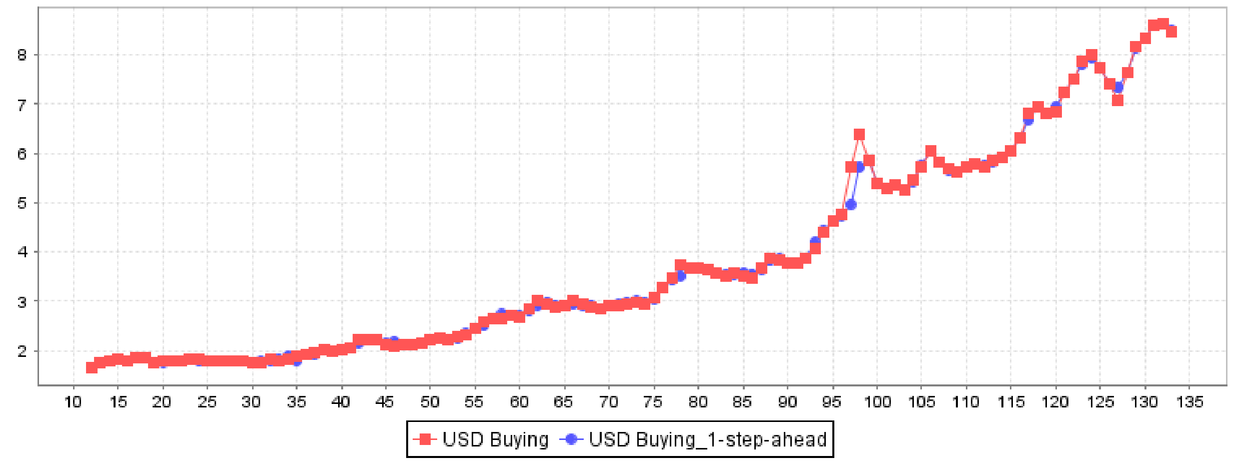

Figure 6. 1step-ahead prediction for USD Buying

The effectiveness of the algorithms was evaluated in this study using the computational results of the forecasting model DA, MAE, MSE and RMSE for time series data on dollar and TL interest rates. The future forecast is shown in Figure 3, while the 1-step forecast is shown in Figure 4. In Figure 5, USD selling is predicted by one step, while in Figure 6, USD buying is predicted by one step ahead. According to these figures, interest in buying and selling USD is steadily increasing. Long-term interest rates in both TL and USD are higher than short-term rates. By and large, the six-month USD rate is the ideal rate. Table 1 shows descriptive statistics for all datasets that illustrate the effectiveness of the learning methods used here, as well as the prediction rates for dollar purchases and sales. When we consider the performance of the algorithms in predicting the change in TRY and USD interest rates, "Linear Regression" yields the best results (100%). The realized estimate for dollar purchases was obtained by Random Forest with 85.53%, MLP with 70.48%, SVM with 84.3% and LWL with 75.06%. For sales in USD, RF achieved 76.95%, LR 100%, MLP 69.22%, SVM 83.47% and LWL 76.85% success. The table also includes the results obtained for various predictive metrics (MAE, RMS, RMSE) and time series. These are additional results of exchange rate estimation. All models show that MAE, RMS and RMSE decrease when sensitivity is included in the model.

Table 1. Evaluation on training data

Algorithms		**1 Month TRY Interest**	**3 Month TRY Interest**	**6 Month TRY Interest**	**1 year TRY Interest**	**1 Month USD Interest**	**3 Month USD Interest**	**6 Month USD Interest**	**1 year USD Interest**	**USD Selling**	**USD Buying**
Random Forest	**MAE**	0.2108	0.2106	0.209	0.1181	0.0345	0.0526	0.0437	0.0288	0.0459	0.0509
	DA	87.5	89.16	88.33	93.33	86.6	88.33	91.26	86.67	**76.95**	**80.53**
	MSE	0.3332	0.3172	0.3371	0.1643	0.047	0.0771	0.0579	0.0401	0.0709	0.0843
	RMSE	0.1111	0.1006	0.1136	0.027	0.0022	0.0059	0.0034	0.0016	0.005	0.0071
Linear Regression	**MAE**	0	0	0	0	0.0797	0	0	0	0	0
	DA	97.5	100	100	98.33	58.43	98.33	96.67	93.33	**100**	**100**
	MSE	0	0	0	0	0.1001	0	0	0	0	0
	RMSE	0	0	0	0	0.01	0	0	0	0	0
Multi-layer Perceptron	**MAE**	0.4353	0.2145	0.3067	0.3508	0.0718	0.102	0.0769	0.0415	0.0932	0.0931
	DA	75	87.5	83.33	82.5	63.33	75	81.67	78.34	**69.22**	**70.48**
	MSE	0.5204	0.2695	0.3646	0.4017	0.091	0.1248	0.0967	0.0562	0.1248	0.1247
	RMSE	0.2708	0.0727	0.1329	0.1613	0.0083	0.0156	0.0093	0.0032	0.0156	0.0156
Support Vector Machine	**MAE**	0.1675	0.1042	0.0645	0.0585	0.0456	0.0411	0.0302	0.0205	0.0348	0.0354
	DA	79.67	89.17	91.66	95	73.34	85.84	83.32	81.66	**83.47**	**84.3**
	MSE	0.3833	0.2474	0.1185	0.129	0.096	0.1188	0.0663	0.0427	0.1016	0.103
	RMSE	0.147	0.0612	0.0141	0.0166	0.0092	0.0141	0.0044	0.0018	0.0103	0.0106
Locally Weighted Learning	**MAE**	0.0397	0.0418	0.0386	0.0386	0.0049	0.0076	0.007	0.0062	0.0286	0.0264
	DA	87.5	94.16	93.33	90.83	93.33	93.34	91.66	89.16	**76.85**	**75.06**
	MSE	0.0599	0.0577	0.0565	0.0562	0.008	0.0116	0.0113	0.0103	0.0405	0.0351
	RMSE	0.0036	0.0033	0.0032	0.0032	0.0001	0.0001	0.0001	0.0001	0.0016	0.0012

4. CONCLUSION

The exchange rate is an important economic indicator of the health of a country's economy. Many elements affect the value of exchange rates, including the balance of international trade, central bank fiscal policy, and exchange rates themselves. Forecasts for the foreign exchange market are critical for investors and companies operating in this sector. According to technical analysis, the activity of a financial market reflects the emotions of investors. The aim of this paper is to build a machine learning model to predict the exchange rate of the Turkish lira against the US dollar using technical analysis. The prediction model has been built based on data from the foreign exchange market, namely the exchange rates of the Turkish lira and the US dollar over certain periods of time. Machine learning techniques such as MLP, LR, SVM, RF and LWL are used to estimate the proposed model. The data was analyzed using MAE, DA, MSE and RMSE. Hence, we believe that our research will benefit risk averse investors by providing basic macroeconomic analysis data that will help investors to make decisions on private equity, venture capital and other illiquid investments.

REFERENCES

[1] M. Štěpnička, P. Cortez, J. P. Donate, and L. Štěpničková, “Forecasting seasonal time series with computational intelligence: On recent methods and the potential of their combinations,” *Expert Syst. Appl.*, vol. 40, no. 6, pp. 1981–1992, 2013, doi: 10.1016/j.eswa.2012.10.001.

[2] M. C. Lee, “Using support vector machine with a hybrid feature selection method to the stock trend prediction,” *Expert Syst. Appl.*, vol. 36, no. 8, pp. 10896–10904, 2009, doi: 10.1016/j.eswa.2009.02.038.

[3] M. Yasir *et al.*, “An intelligent event-sentiment-based daily foreign exchange rate forecasting system,” *Appl. Sci.*, vol. 9, no. 15, 2019, doi: 10.3390/app9152980.

[4] D. Shah, H. Isah, and F. Zulkernine, “Stock market analysis: A review and taxonomy of prediction techniques,” *Int. J. Financ. Stud.*, vol. 7, no. 2, 2019, doi: 10.3390/ijfs7020026.

[5] M. J. S. de Souza, D. G. F. Ramos, M. G. Pena, V. A. Sobreiro, and H. Kimura, “Examination of the profitability of technical analysis based on moving average strategies in BRICS,” *Financ. Innov.*, vol. 4, no. 1, 2018, doi: 10.1186/s40854-018-0087-z.

[6] F. Cavalli, A. Naimzada, and M. Pireddu, “An evolutive financial market model with animal spirits: imitation and endogenous beliefs,” *J. Evol. Econ.*, vol. 27, no. 5, pp. 1007–1040, 2017, doi: 10.1007/s00191-017-0506-8.

[7] J. Contreras, R. Espínola, F. J. Nogales, and A. J. Conejo, “ARIMA models to predict next-day electricity prices,” *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1014–1020, 2003, doi: 10.1109/TPWRS.2002.804943.

[8] T. Bollerslev, “Generalized autoregressive conditional heteroskedasticity,” *J. Econ.*, vol. 31, pp. 307–327, 1986, doi: 10.3905/jpm.2019.1.098.

[9] Q. H. Luu, M. F. Lau, S. P. H. Ng, and T. Y. Chen, “Testing multiple linear regression systems with metamorphic testing,” *J. Syst. Softw.*, vol. 182, p. 111062, 2021, doi: 10.1016/j.jss.2021.111062.

[10] M. Khashei and M. Bijari, “An artificial neural network (p, d, q) model for timeseries forecasting,” *Expert Syst. Appl.*, vol. 37, no. 1, pp. 479–489, 2010, doi: 10.1016/j.eswa.2009.05.044.

[11] S. Galeshchuk, “Neural networks performance in exchange rate prediction,” *Neurocomputing*, vol. 172, pp. 446–452, 2016, doi: 10.1016/j.neucom.2015.03.100.

[12] B. Amrouche and X. Le Pivert, “Artificial neural network based daily local forecasting for global solar

radiation," *Appl. Energy*, vol. 130, no. 2014, pp. 333–341, 2014, doi: 10.1016/j.apenergy.2014.05.055.

[13] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," *J. Pharm. Biomed. Anal.*, vol. 22, no. 5, pp. 717–727, 2000, doi: 10.1016/S0731-7085(99)00272-1.

[14] "Investing in Turkey, 2010-2021 | Kaggle." https://www.kaggle.com/captainozlem/investing-in-turkey-2010-2021/data (accessed Dec. 01, 2021).

[15] M. Lam, "Neural network techniques for financial performance prediction : integrating fundamental and technical analysis," vol. 37, pp. 567–581, 2004, doi: 10.1016/S0167-9236(03)00088-5.

[16] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java." https://www.cs.waikato.ac.nz/ml/weka/ (accessed Dec. 13, 2021).

[17] Z. Jin, J. Shang, Q. Zhu, C. Ling, W. Xie, and B. Qiang, "RFRSF: Employee Turnover Prediction Based on Random Forests and Survival Analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12343 LNCS, pp. 503–515, 2020, doi: 10.1007/978-3-030-62008-0_35.

[18] "Multilayer Perceptrons - an overview | ScienceDirect Topics." https://www.sciencedirect.com/topics/computer-science/multilayer-perceptrons (accessed Dec. 02, 2021).

[19] M. A. Ghorbani, R. C. Deo, Z. M. Yaseen, M. H. Kashani, and B. Mohammadi, "Pan evaporation prediction using a hybrid multilayer perceptron-firefly algorithm (MLP-FFA) model: case study in North Iran," *Theor. Appl. Climatol.*, vol. 133, no. 3–4, pp. 1119–1131, 2018, doi: 10.1007/s00704-017-2244-0.

[20] M. A. Ghorbani, R. Khatibi, B. Hosseini, and M. Bilgili, "Relative importance of parameters affecting wind speed prediction using artificial neural networks," *Theor. Appl. Climatol.*, vol. 114, no. 1–2, pp. 107–114, 2013, doi: 10.1007/s00704-012-0821-9.

[21] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[22] P. Rivas-Perea, J. Cota-Ruiz, D. G. Chaparro, J. A. P. Venzor, A. Q. Carreón, and J. G. Rosiles, "Support Vector Machines for Regression: A Succinct Review of Large-Scale and Linear Programming Formulations," *Int. J. Intell. Sci.*, vol. 03, no. 01, pp. 5–14, 2013, doi: 10.4236/ijis.2013.31002.

[23] Y. Alqasrawi, M. Azzeh, and Y. Elsheikh, "Science of Computer Programming Locally weighted regression with different kernel smoothers for software effort estimation," vol. 214, pp. 1–17, 2022.

[24] S. Schaal, C. G. Atkeson, and S. Vijayakumar, "Real-time robot learning with locally weighted statistical learning," *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 1, no. March 2015, pp. 288–293, 2000, doi: 10.1109/robot.2000.844072.

[25] N. Mehdiyev, D. Enke, P. Fettke, and P. Loos, "Evaluating Forecasting Methods by Considering Different Accuracy Measures," *Procedia Comput. Sci.*, vol. 95, pp. 264–271, 2016, doi: 10.1016/j.procs.2016.09.332.

[26] A. A. Syntetos and J. E. Boylan, "The accuracy of intermittent demand estimates," *Int. J. Forecast.*, vol. 21, no. 2, pp. 303–314, 2005, doi: 10.1016/j.ijforecast.2004.10.001.

[27] T. M. Usha and S. A. A. Balamurugan, "Seasonal Based Electricity Demand Forecasting Using Time Series Analysis," *Circuits Syst.*, vol. 07, no. 10, pp. 3320–3328, 2016, doi: 10.4236/cs.2016.710283.

[28] I. Moosa and J. Vaz, "Directional accuracy, forecasting error and the profitability of currency trading: model-based evidence," *Appl. Econ.*, vol. 47, no. 57, pp. 6191–6199, 2015, doi: 10.1080/00036846.2015.1068917.

[29] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? -Arguments against avoiding RMSE in the literature," *Geosci. Model Dev.*, vol. 7, no. 3, pp. 1247–1250, 2014, doi: 10.5194/gmd-7-1247-2014.